Mathematical modeling of tumor and metastatic growth when treated with sunitinib

Simon Evain, Sebastien Benzekry

Internship report

Mathematical modeling of tumor and metastatic growth when treated with sunitinib

Simon Evain

supervised by
Sébastien Benzekry

October 8, 2016

Abstract

The effect of the antiangiogenic drug sunitinib on the tumor size and the metastatic burden evolution are studied statistically and mathematically. For this, a new model SunitinibTGI is built, which allows to model with accuracy the phenomenon, thus giving us more information on the behavior of a tumor treated with sunitinib, distinguishing a phase of resistance to sunitinib under certain conditions. Together with some correlations that are noted on the data set, these tools allow us to better understand the metastatic response to sunitinib administration, notably in a neoadjuvant setting.

Contents

List of Figures

1 Introduction

Cancer is, in France, the first cause of mortality ([12]). In spite of the constant progress of medical research, cancer is still an illness which is poorly understood in a lot of aspects.

Cancer is a disease which is born when some cells of the organism break free from the genetic rules and undergo limitless mitosis. This leads to the creation of a mass of cells with uncontrollable growth within the organs, which have no utility whatsoever for the organism: the primary tumor. At some stage, the tumor cannot grow more due to the lack of nutrients. But some primary tumors find themselves able to induce what is called angiogenesis, which allows to stimulate its own vascularization. This way, the tumor can access more nutrients and grow even more.

Some tumor cells may be able to escape the organ, through the blood or the lymph nodes, giving the possibility of a dissemination of tumor cells inside the whole organism. These cells are then able to settle in a new organ (colonization), and to grow inside the organ to form a secondary tumor. This latest process is called metastatic process, and is the least understood and the most lethal part in most cancers.

One of the possible ways to treat cancer is therefore to block angiogenesis, in order to deprive the tumor from its nutrients, thus weakening it. Among the most well-known antiangiogenic drugs, we can quote the Sutent (or sunitinib, for the name of the molecule), which has recently been the object of a controversy. One of the main goals for the administration of such a medicine is the size reduction of the primary tumor, allowing to remove the tumor through surgery much easier later. And if, indeed, this specific consequence has been evidenced by several experiments, John Ebos, in [6], has made a connection between the administration of the drug and the metastatic acceleration for mice. Administering such a drug could lead to an acceleration of the metastatic process for mice, and ultimately reducing their life expectancy.

During this six-month-long internship carried out at inria Bordeaux, within the monc team, the main objective was to study the connection between these two factors, through the analysis of Mr John Ebos' data. The first objective was to try and describe as accurately as possible the evolution of the size of the primary tumor in those experiments, when treated and untreated. A benchmark of classical models had to be performed, to compare their relative efficiency to fit the data. The fitting process itself had to be studied in order to pick the best (most stable, accurate and quicker) optimization algorithm. Faced with rather unsatisfying results when fitting the

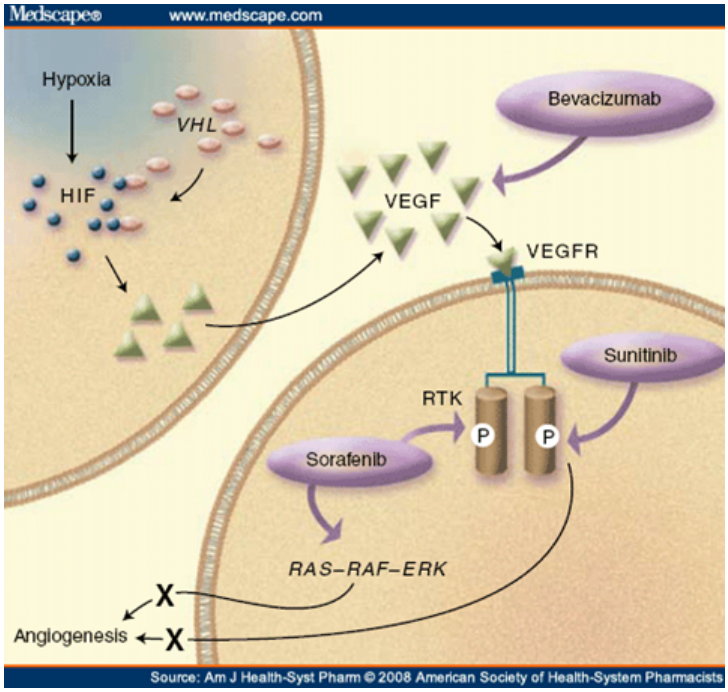


Figure 1: *Description of the effect of sunitinib*

treated data to classical methods, a new model was built to try and comprehend the evolution of the disease. From this model, called SunitinibTGI, that we proved to be more efficient statistically than the classical models for our data (and with identifiable parameters), we were also able to learn more things on the primary tumor size evolution. Besides, performing correlations on several amounts of data showed some interesting results.

Incorporating this new primary tumor size evolution model into a more classical metastatic process equation allowed us to check whether the metastatic acceleration was indeed notable or somewhat insignificant mathematically.

The report that follows will, for that reason, first discuss the biological point of view more specifically. Then, the various benchmarks performed will be presented, which will justify the creation of a new model ; its properties will then be studied. Finally, correlations and the metastatic growth will be discussed.

It should be noted that 4 annexes are present within the report: an annex devoted to cancer biology, an annex devoted to the statistical tools that were employed during the internship, an annex related to the optimization benchmark that was carried out, and finally an annex devoted to the nu-

merical methods employed. All the code used for the internship was written using Matlab, and the internship was an extension of the work carried out previously by Mr Aristoteles Camillo ([12]).

2 Description of the problem

2.1 Nature of the issue

2.1.1 A brief summary on sunitinib

Sunitinib (marketed by Pfitzer as Sutent) is a drug that was approved in 2006 by the Food and Drug Administration simultaneously for two diseases: metastatic kidney cancer, and GIST (Gastro-Intestinal Stromal Tumor) as a second-line drug (after the potential failure of imatinib). This drug acts like a targeted bio-therapy: it does not attack straightly the cells, but attacks instead a specific biological action mechanism used for the growth of tumor cells. That action mechanism is angiogenesis, the biological process by which new blood vessels are created from pre-existing ones, allowing the tumor to grow beyond a certain size to be able to eventually metastasize.

Sunitinib acts as an inhibitor for Tyrosine Kinase Receptors, the enzymes which receive the signals emitted by the primary tumor allowing the further growth of the vasculature within the primary tumor. Among the receptors that are inhibited by the drug are the VEGFRs (Vascular Endothelial Growth Factor Receptors) and PDGFRs (Platelet-Derived Growth Factor Receptors).

Besides, sunitinib is a small molecule, and is therefore able to enter into the cells, unlike most antibodies.

2.1.2 Controversy on the effect of sunitinib in a neo-adjuvant setting

In 2009, the American biologist John Ebos and his associated co-workers published an article [6], that was seen as a major breakthrough. In this article, Ebos showed that a metastatic acceleration could occur for a mouse treated with sunitinib in a neo-adjuvant perspective (before tumor resection). These results seem to indicate that a treatment of sunitinib in a neo-adjuvant setting can reduce the life expectancy of mice.

This matter of the exact effect of sunitinib on tumor and metastatic kinetics is absolutely crucial, for it is a very common and widespread antiangiogenic drug. For that reason, other studies were led on human patients, which tended to show that there was no clear metastatic acceleration and

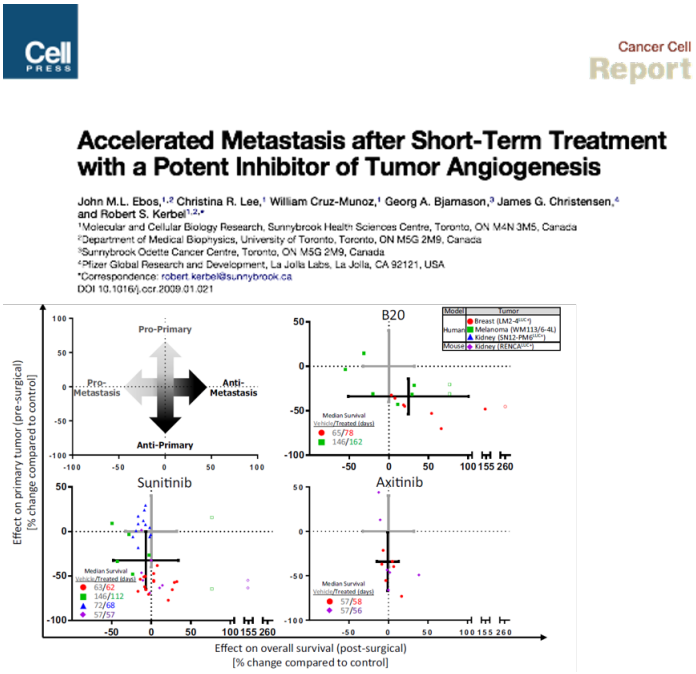


Figure 2: *Presentation of the 2009 article by J. Ebos et al.*

overall survival reduction when administered with sunitinib.

This issue was central in the internship.

2.2 Data description

All the data this internship is built upon were provided by Dr Ebos, the American biologist who first assumed the idea of connection between sunitinib administration and metastatic acceleration.

The data is extracted from mice, and all are drawn from two different cell lines: breast and melanoma, both treated with sunitinib.

The breast data are data on primary tumor size and, for some groups, metastatic burden. 88 mice were subject to the experiment, and, among them, 33 were used as control individuals (so they were injected with the cancer cells, but not with the treatment). Six various groups, differing by their scheduling, can be distinguished. For four out of six groups, the primary tumor was resected, allowing to evaluate the evolution of the metastatic burden when sunitinib is administered in a neo-adjuvant perspective. For the two remaining groups (31 and 32), the primary tumor was not resected ; these groups are useful for they permit to study tumor growth kinetics on

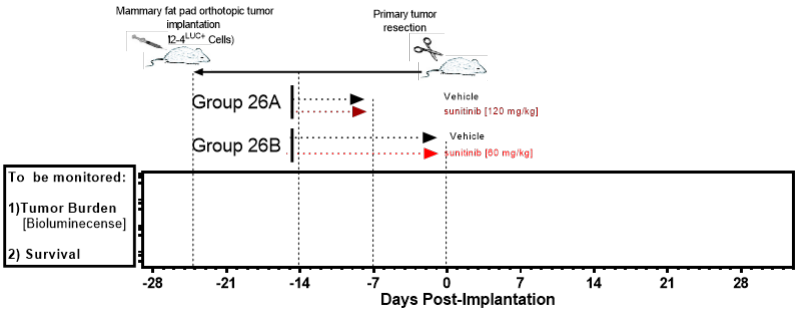


Figure 3: *Description of the experimental process for a group of mice*

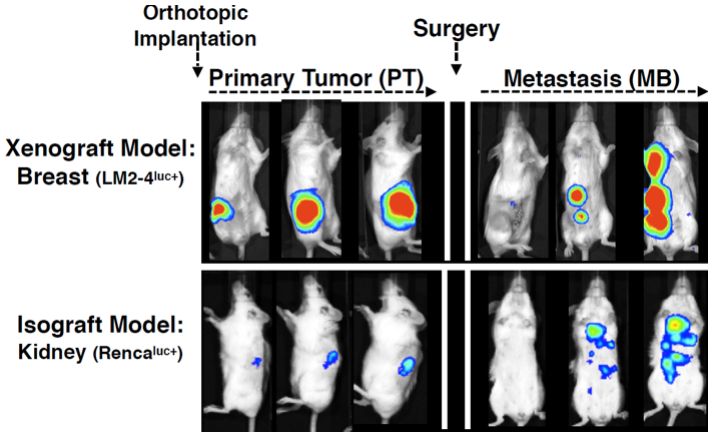


Figure 4: *Description of the experimental process*

Group number	Dose injected ($mg.kg^{-1}$)	Treatment days	Resection day
1	120	24-30	32
2	60	24-38	39
31	60	11-60	-
32	120	2-19	-
51	60	20-33	34
6	60	20-33	34

Figure 5: *Summary of all breast-related data that were dealt with during the internship*

Group number	Drug employed	Cell line	Treatment days
1A	516	regular	1-41
1B	sunitinib	regular	1-41
2A	516	resistant	1-41
2B	sunitinib	resistant	1-41

Figure 6: *Summary of all melanoma-related data that were dealt with during the internship*

a longer scale. Figure 5 displays the different treatments for each group.

Another data set corresponds to melanoma-related cells. 38 mice are subject to such an experiment, among which 22 are exploited for control analysis (so, untreated) and the 16 remaining mice can be divided into 4 groups. All these data have in common the fact that they only show primary tumor growth. Besides, all tumors are treated from day 1 to day 41, and the treatment is then stopped. These pieces of data, thus, allow to evaluate the impact of a stop of the treatment. Two various drugs (516 and sunitinib) were used on two different cell lines (regular or sunitinib-resistant cells). All the useful information regarding this data set are indicated on figure 6.

All primary tumor growth volume data are expressed in mm^3. The volume is calculated thanks to the formula: $V = 0.5 width^2.length$, where length and width of the tumor cells are measured using a caliper. The metastatic burden is measured using bio-luminescence, and is therefore expressed in photons per second.

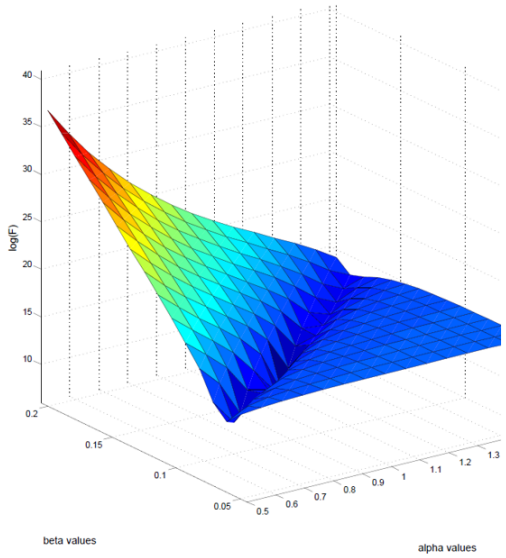


Figure 7: *Surface of the objective function to be optimized for a Gompertz function*

2.3 Benchmark of optimization algorithms

Our objective during this internship is to model the evolution of tumor growth. On a mathematical point of view, we need to solve the following least-square minimization problem (y_i designates the data, p the parameters of the model and M the chosen model):

$$\min_p(\sum_i(y_i - M(t_i; p))^2)$$

To address this issue, a test case was carried out on a Gompertz model and the data for mice which were not administered any treatment.

Several algorithms and methods were implemented to be compared. Were tested the constant gradient, variable gradient, Gauss-Newton and Levenberg-Marquardt algorithms. The results are described in the annex devoted to Optimization in this report, but only its outcome will be displayed here.

We concluded that the Nelder-Mead and the Levenberg-Marquardt algorithms were to be favoured in terms of stability and efficiency.

One important point that needs to be understood is that, no matter how accurate our optimization process is, there is still a huge influence of the initial condition. For that reason, for each data fitting that will be

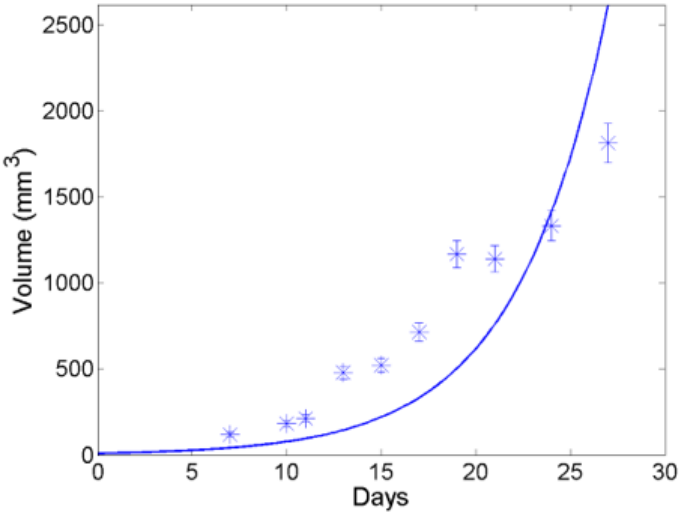


Figure 8: *Fit of the average data of the group 3 by the exponential model*

performed in the subsequent parts of the report, the optimization algorithm was launched a significant number of times, with various starting points.

3 Classical models of tumor growth without treatment ([1])

The first data that needs to be fitted with models is the evolution of the size of the primary tumor of mice for control groups (so when the mice are not treated). For that reason, all the following models were implemented on MATLAB, and the Nelder-Mead algorithm was used each time, on the objective function described in the previous part. This allowed us to find the parameters of each model which were able to describe as accurately as possible the evolution of the primary tumor. We then computed the result that was obtained to see how close it was from the actual data. This way, we were able to compare all the following models.

3.1 Exponential model

Tumor growth in this model is supposed to be exponential.

$$\frac{dV}{dt} = \lambda V \tag{1}$$

where λ is the growth rate of the primary tumor.

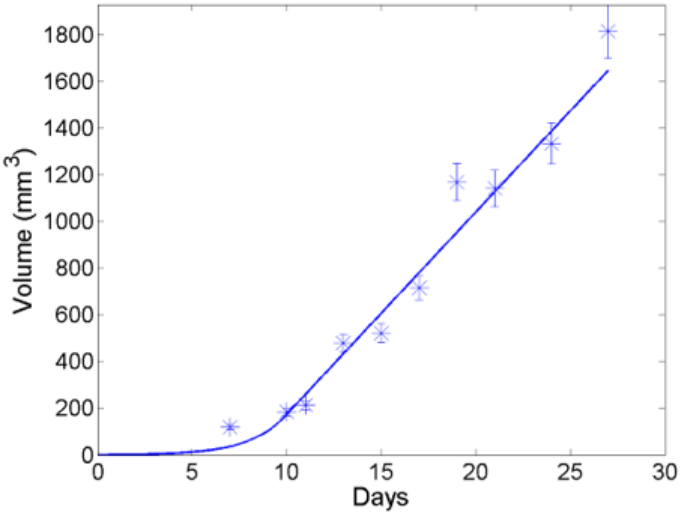


Figure 9: *Fit of the average data of the group 3 by the exponential-linear model*

3.2 Exponential-linear model

This model takes into account more elements, when compared with the exponential model. It follows 2 successive phases:

- an exponential growth with a constant rate a_0.
- a linear growth with a constant rate a_1.

$$\begin{cases} \frac{dV}{dt} = a_0 V(t), & t \leq \tau \\ \frac{dV}{dt} = a_1, & t \geq \tau \\ V(t=0) = V_0 \end{cases} \tag{2}$$

To use the model, we often resort to a single differential equation, which approximates the behavior described by the previous equations:

$$\frac{dV}{dt} = \frac{a_0 V(t)}{(1 + (\frac{a_0 V(t)}{a_1})^{\psi})^{\frac{1}{\psi}}} \tag{3}$$

ψ is often chosen equal to 20 (value taken from [1]). We notice that when V is small, the growth of the model is very close to the exponential growth

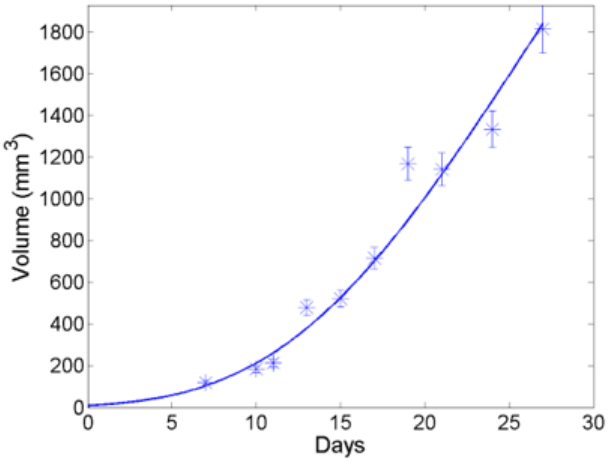


Figure 10: *Fit of the average data of the group 3 by the Gompertz model*

; when V is large, the model gets closer to a linear one.

Besides, if we want the solution to be continuously differentiable, we see that we necessarily have $\tau = \frac{1}{\log(\frac{a1}{a_0 V_0})}$.

On a biological point of view, the model represents the fact that the growth of the tumor is almost unlimited at first, but that as the nutrients go scarce, its growth will then necessarily be restricted, and become linear.

3.3 Gompertz model

In this model, tumor growth is written this way:

$$\frac{dV}{dt} = (a - b\ln(V))V \tag{4}$$

In the equation, a is the tumor proliferation rate, and b is the rate of exponential decay of the proliferation rate.

Biologically, the model is based on the same reasoning as the exponential-linear model, but with the supplementary assumption that the tumor size has an upper boundary: the carrying capacity $V_0 \exp(\frac{a}{b})$. It cannot grow until infinity.

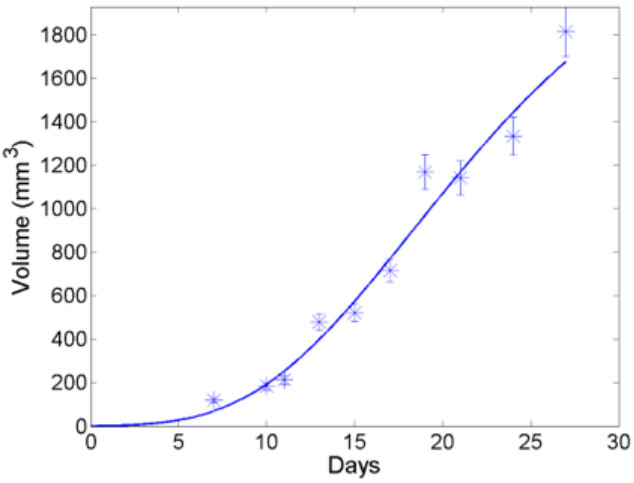


Figure 11: *Fit of the average data of the group 3 by the Gomp-Exp model*

3.4 Gomp-Exp model

The Gomp-Exp model is built upon the following idea: at first, when all nutrients are available, the tumor follows an exponential growth. When the nutrients start to go scarce, then the tumor growth is akin to a Gompertz model. This can be formalized as:

$$\frac{dV}{dt} = \min(\lambda V, (a - b\ln(V))V) \tag{5}$$

3.5 Power Law model

A way to express tumor growth is through what is called the Power Law model, which can be written this way:

$$\frac{dV}{dt} = aV^{\gamma}(t) \tag{6}$$

In this equation, a represents the proliferation rate of the tumor, supposed to be the same for all same cell lines, and γ is related to its vasculature of the tumor in the current situation. Since we are specifically studying the angiogenic process, and that we will try and model the vascularization of the tumor, we can see that, independently from its performances, this model has parameters that are extremely relevant to the subject that we are working on.

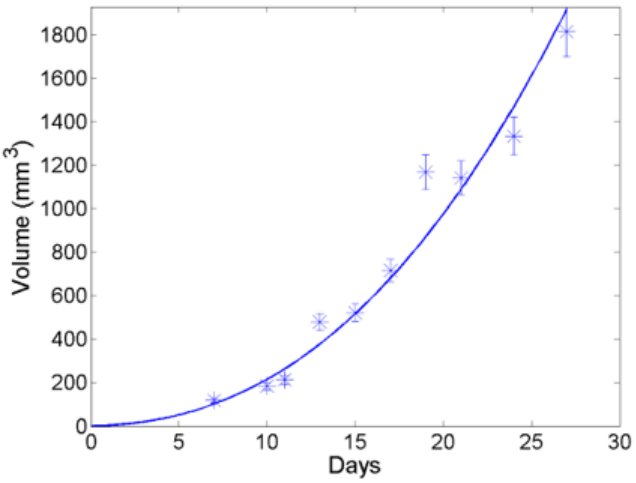


Figure 12: *Fit of the average data of the group 3 by the Power Law model*

3.6 Logistic model

In this model, the carrying capacity K is considered as a constant in the problem. We can then write the model as:

$$\frac{dV}{dt} = aV(t)(1 - \frac{V(t)}{K}) \tag{7}$$

In this expression, a is considered as a rate controlling the velocity of the growth. In this model, as the tumor volume reaches the carrying capacity, the growth is gradually reduced.

3.7 Dynamic CC model

This model considers the carrying capacity K as a variable representing the tumor vasculature. The dynamic can then be written as:

$$\begin{cases} \frac{dV}{dt} = aV(t)\log(\frac{K(t)}{V(t)}) \\ \frac{dK}{dt} = bV(t)^{\frac{2}{3}} \\ V(t=0) = V_0, K(t=0) = K_0 \end{cases} \tag{8}$$

The biological reasoning behind such a model is close from the reasoning guiding the other models. We can note that the factor $\frac{2}{3}$ is chosen because

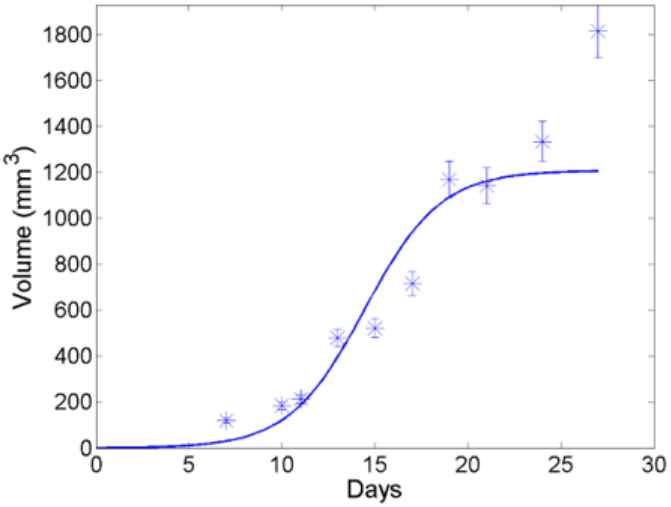


Figure 13: *Fit of the average data of the group 3 by the Logistic model*

it is assumed that the carrying capacity growth is proportional to the tumor surface.

3.8 Comparison of the accuracy of the untreated growth models

We implemented all the previous models and fit the data from control groups with them, using the Nelder-Mead algorithm as the optimization method. The parameters are the result of the optimization problem of minimization of the SSE. The results for this statistical comparison are summed up on figure 14 (the meaning of each one of the statistical tools employed are described in the Annex Statistics).

For the statistical tools SSE, AIC and RMSE, it is more interesting to have a small value (see the Annex Statistics to understand why) ; as for the R^2, the closer it is to 1, the better the actual fit is.

We notice that for all criteria, two models seem to have to be favored: the Gompertz model and the Power Law model. We can note that since our work aims at modeling the effect of anti-angiogenic drugs, the biological meaning of the parameters related to the Power Law model, notably through their modeling of vascularization, could be quite useful.

Goodness of fit

Model	SSE	AIC	RMSE	R2	p > 0.05	#
Gompertz	**0.144(0.0851 - 0.235)[1]**	**-13.9(-18.6 - -8.49)[1]**	**0.393(0.306 - 0.509)[1]**	**0.957(0.9 - 0.992)[1]**	100	2
Power law	0.146(0.0883 - 0.222)[2]	-13.7(-18.3 - -9.05)[2]	0.397(0.313 - 0.493)[2]	0.952(0.911 - 0.991)[2]	100	2
Gomp-Exp	0.193(0.104 - 0.33)[3]	-11.2(-16.6 - -5.08)[3]	0.448(0.331 - 0.592)[3]	0.934(0.853 - 0.975)[3]	100	2
Exponential-linear	0.283(0.175 - 0.394)[4]	-7(-11.4 - -3.31)[4]	0.547(0.433 - 0.653)[4]	0.879(0.767 - 0.966)[4]	100	2
DynaCC	0.35(0.0994 - 0.889)[5]	-5.22(-15.1 - 6.83)[5]	0.57(0.332 - 0.952)[5]	0.868(0.57 - 0.99)[5]	100	3
Logistic	0.674(0.46 - 1.05)[6]	1.68(-1.77 - 6.45)[6]	0.826(0.688 - 1.03)[6]	0.758(0.621 - 0.879)[6]	100	2
Exponential	1.49(0.811 - 2.01)[7]	7.48(1.9 - 11)[7]	1.21(0.902 - 1.42)[7]	0.567(0.213 - 0.79)[7]	0	1

Figure 14: *Comparison of the accuracy of the untreated growth models*

4 Implementation of the treatment in a tumor growth model

4.1 Shape of the treated data

Before performing any analysis, it is important to focus on the treated tumor data (volume) graphically. We can separate all the data set into 3 different kinds of behavior, 2 for breast-related data and a specific behavior for melanoma-related data.

'Behavior breast 1', as displayed in figure 15, shows a tumor growth which can be distinguished into 3 phases.

- Phase 1 (pre-treatment): the tumor grows and no treatment is administered yet. We can consider this growth as identical to the tumor growth for control groups, when absolutely no treatment is administered.

- Phase 2 (growth arrest): this phase begins when the treatment starts to be administered, and it lasts for a certain duration which does not seem to always coincide with the duration of the treatment. During

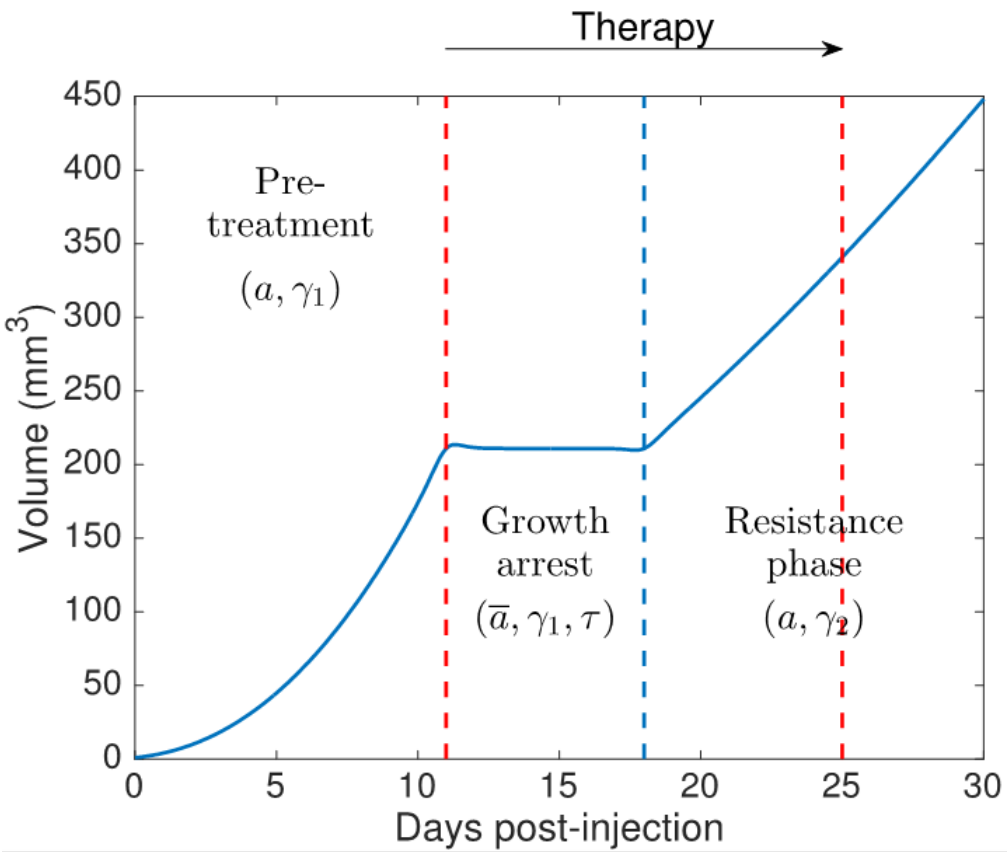


Figure 15: *Shape of the treated data for breast cancer ['behavior breast 1 and 2']*

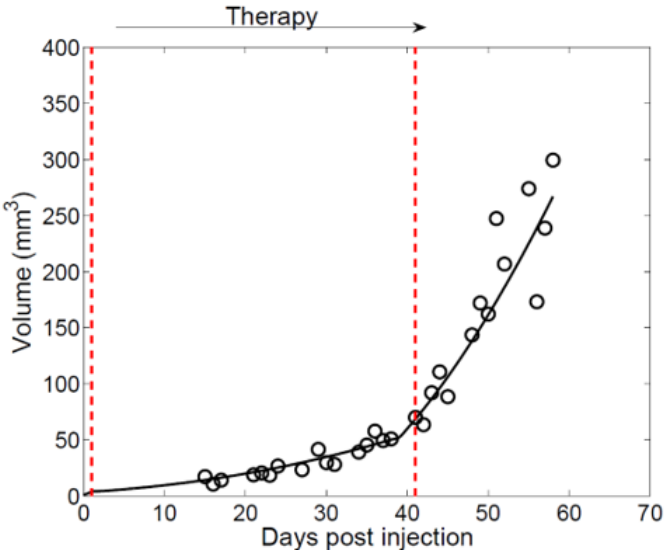


Figure 16: *Shape of the treated data for melanoma ['behavior melanoma']*

this phase, the treatment seems to be extremely efficient, and to stop nearly all tumor growth.

- Phase 3 (resistance phase): after a certain duration, the tumor regrows, and the treatment, while still administered at the same dose, seems to be far less effective.

'Behavior breast 2' corresponds to the same data, but for mice that had their tumor resected sooner, and they then only exhibit the first two phases of the previous behavior.

The third one, 'behavior melanoma', can be divided into two phases. First, there is no pre-treatment phase, for the treatment begins the day after the tumor is implanted. Then, we notice in the data a first phase of tumor growth, which seems to last for about the same duration as the administration of the treatment. A second phase seems to occur when the treatment stops, with a tumor growth that is accelerated. But these observations, as for the exact duration of this phase, need to be verified mathematically and statistically.

4.2 Approach based on pharmacokinetics ([15])

Most models in the literature are built upon the idea of pharmacokinetics. Here, in this specific framework, we can see the whole human body, and in

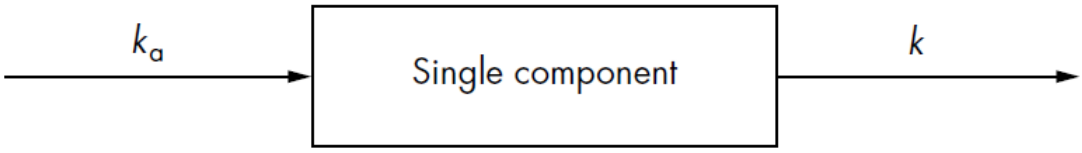


Figure 17: *Unicompartmental model (absorption rate k_a and elimination rate k*

particular the blood system, as a compartment, as shown in figure 17 (unicompartmental model with absorption). First, the drug enters the body, its amount is given by a variable X_a. The drug reaches the gut, in which an absorption occurs, the absorption being defined by the variable k_a. The drug is then spread in the body, and is eliminated, the elimination being modeled by a variable k. We call C_{AA} the concentration of drugs actually in the body, and we call $T_{admin}(m)$, the times at which the drug is administered, at a dose D_m. Besides, we call V_1 the fictive volume of the compartment.

We have the following equations:

$$\begin{cases} \frac{dX(t)}{dt} = k_a X_a(t) - kX(t) \\ \frac{dX_a(t)}{dt} = -k_a X_a(t) \end{cases}$$

First, let us suppose that the drug is only administered once at a time T_{adm} and a dose D. We obtain $X_a(t) = X_{0a}\exp(-k_a(t - T_{adm}))$. Which leads us to

$$\frac{dX}{dt} = (k_a X_{0a}\exp(-k_a(t - T_{adm}))).(t > T_{adm}).$$

Using the variation of constants, we finally find

$$X(t) = D\frac{k_a}{k_a - k}(\exp(-k(t - T_{adm})) - \exp(-k_a(t - T_{adm}))).(t > T_{adm}).$$

Now, if we suppose that we administer the drug at times $T_{admin}(m)$, every time with the dose D_m, we obtain the following result:

$X(t) = \sum_m D_m \frac{k_a}{k_a - k}(\exp(-k(t - T_{adm}(m))) - \exp(-k_a(t - T_{adm}(m)))).(t > T_{adm}(m))$.

Since X(t) is the amount of drug in the body, we have $X(t) = C_{AA}(t)V_1$.

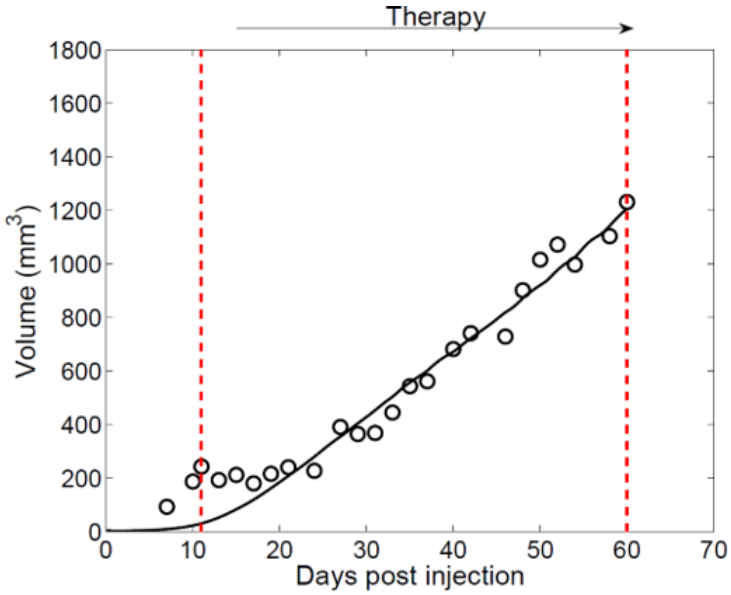


Figure 18: *Fit of the average data of the group 31 by the Exp-log kill model*

$c_{AA}(t) = \sum_m \frac{D_m}{V_1} \frac{k_a}{k_a - k}(\exp(-k(t - T_{adm}(m))) - \exp(-k_a(t - T_{adm}(m)))).(t > T_{adm}(m))$.

In what will ensue, c_{AA} will be calculated this way.

4.3 Classical models of tumor growth with treatment

The optimization of the objective function described in the "Optimization" subsection is once more computed, but with a different data set (treated tumor) and with new models, supposedly more able to describe the data.

4.3.1 Exp-log kill

The Exp-log kill model can be expressed with the following equation:

$$\frac{dV}{dt} = (\alpha - \beta c_{AA}(t))V(t) \tag{9}$$

The idea behind this model is that the natural tumor growth (illustrated by the parameter α) is compensated by the effect of the treatment (parameter β). It can be noted that the effect of the treatment is assumed to be proportional to the drug concentration in the organism and to the tumor volume (assumed to be proportional to the total number of tumor cells).

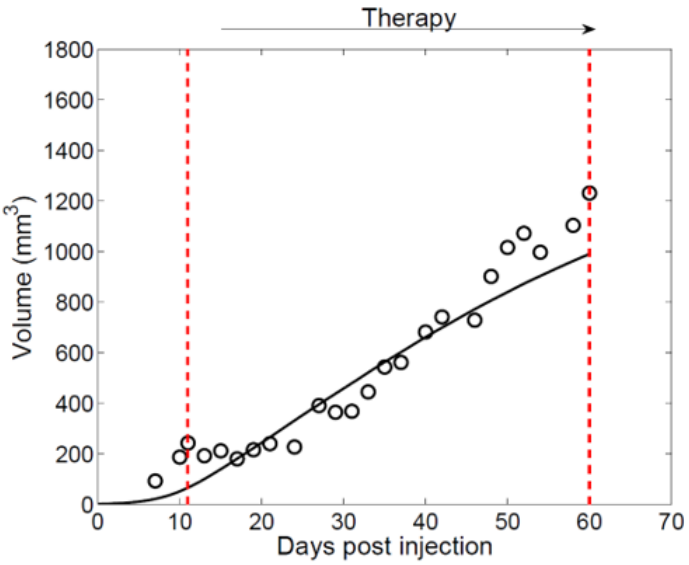


Figure 19: *Fit of the average data of the group 31 by the Gompertz-Norton-Simon model (3 degrees of freedom)*

4.3.2 Gompertz-Norton-Simon model

The Gompertz-Norton-Simon model can be written as:

$$\frac{dV}{dt} = (\alpha - \beta \ln(V(t)))V(t)(1 - ec_{AA}(t)) \tag{10}$$

It can be noted that this model is an adaptation of the Gompertz model used for control groups, where a simple pharmacokinetic term was added, partly determined by the parameter e, which needs to be fitted as a supplementary parameter.

4.3.3 SimeoniAA model

This model is an adaptation of the Simeoni model used for control data. It can be expressed as:

$$\frac{dV}{dt} = \frac{a_0 V(t)}{(1 + (\frac{a_0}{a_1} V(t))^{\psi})^{\frac{1}{\psi}}}(1 - \frac{c_{AA}(t)}{c_{AA}(t) + IC_{50}}) \tag{11}$$

The pharmacokinetics is this time implemented by a somewhat different term, which allows once more to evaluate the drug effect through its concentration in the organism (IC_{50}, in this model, corresponds biologically to the required drug concentration to halve tumor growth).

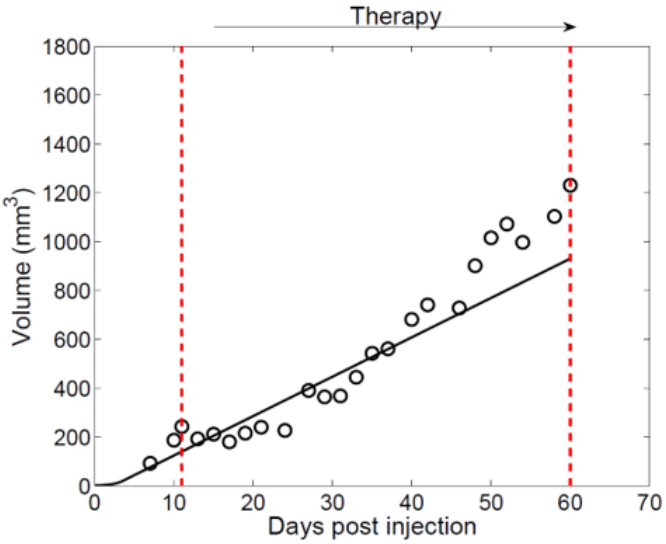


Figure 20: *Fit of the average data of the group 31 by the SimeoniAA model (3 degrees of freedom)*

4.3.4 Simeoni2004 model

This model, first expressed in [14], is a more sophisticated model to try and model tumor growth which adopts the following form (once again adapted from the Simeoni model in control groups):

$$\begin{cases} \frac{dx_1}{dt} &= \frac{a_0 x_1(t)}{[1+(\frac{a_0}{a_1}V(t))^{\psi}]^{\frac{1}{\psi}}} - k_2 c_{AA}(t) x_1(t) \\ \frac{dx_2}{dt} &= k_2 a(t) x_1(t) - k_1 x_2(t) \\ \frac{dx_3}{dt} &= k_1[x_2(t) - x_3(t)] \\ \frac{dx_4}{dt} &= k_1[x_3(t) - x_4(t)] \\ V(t) &= x_1(t) + x_2(t) + x_3(t) + x_4(t) \end{cases}$$

This model results from a pharmacokinetic multi-compartmental approach of the problem (4 compartments), analogous to what was described in the 'Approach based on pharmacokinetics' subsection of the report. Here, the primary tumor volume is considered as the sum of four different tumor cells: x_1 indicates the portion of proliferating cells within the total tumor volume, and the other functions x_2, x_3 and x_4 represent the non-proliferating cells at later stages.

4.3.5 Treated dynamic CC model

This model is adapted from the dynamic CC model used for the fitting of untreated data. It can be written as the following equations:

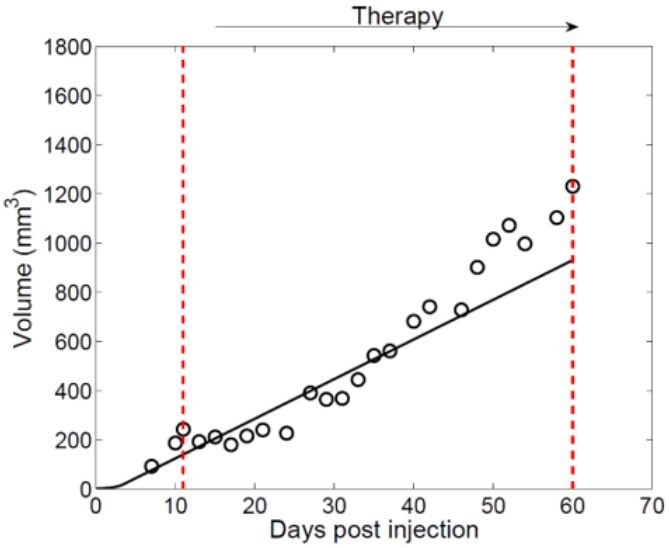


Figure 21: *Fit of the average data of the group 31 by the Simeoni2004 model (4 degrees of freedom)*

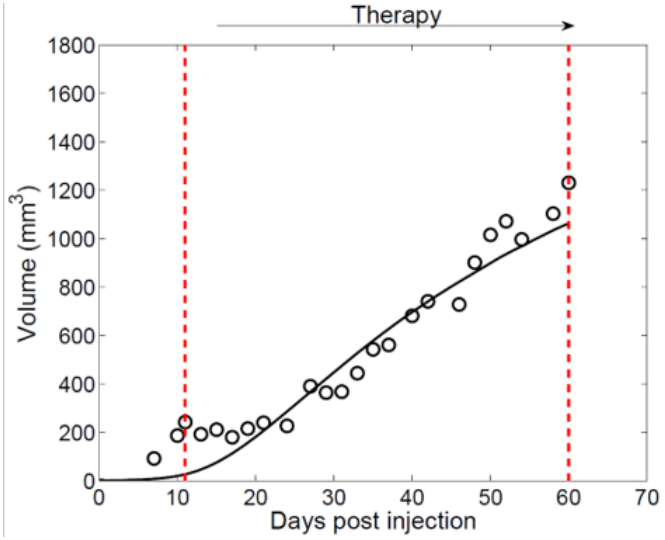


Figure 22: *Fit of the average data of the group 31 by the dynamic CC model (2 degrees of freedom)*

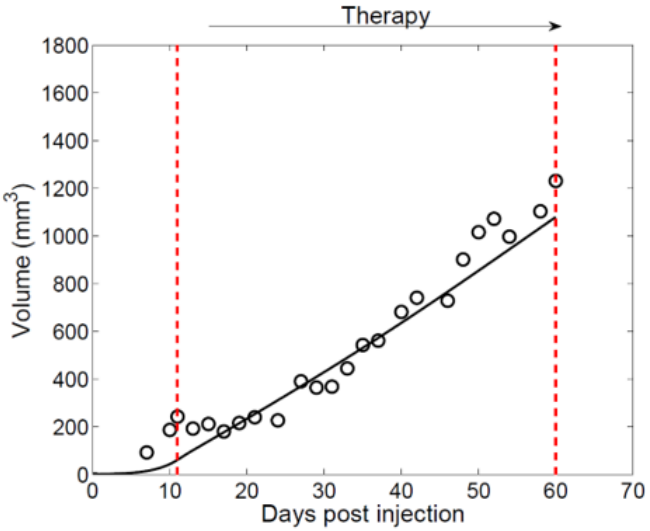


Figure 23: *Fit of the average data of the group 31 by the Hahnfeldt model (3 degrees of freedom)*

$$\begin{cases} \frac{dV}{dt} = aV(t)\ln(\frac{K(t)}{V(t)}) \\ \frac{dK}{dt} = aV(t)^{\frac{2}{3}} - ec_{AA}(t)V(t) \end{cases}$$

As a reminder, K represents here the carrying capacity (upper boundary of tumor size), which is taken as a variable. We note that the pharmacokinetic adaptation of the model considers the effect of the treatment as a term which reduces the carrying capacity of the tumor. This last statement makes sense biologically, for the antiangiogenic effect of sunitinib is indeed supposed to reduce the vasculature of the tumor and thus the carrying capacity in this model.

4.3.6 Hahnfeldt model

This model was first pitched in [17].

$$\begin{cases} \frac{dV}{dt} = aV(t)\ln(\frac{K(t)}{V(t)}) \\ \frac{dK}{dt} = bV(t) - dV(t)^{\frac{2}{3}} - ec_{AA}(t)K(t) \end{cases}$$

The differences are notable in the evolution of the carrying capacity. Indeed, unlike the treated dynamic CC model, the influence of tumor volume on carrying capacity growth is decomposed into two phenomena (birth and death) illustrated by two distinct parameters (b for birth rate and d for death rate). Besides, the effect of the treatment on the carrying capacity is this time proportional to the carrying capacity instead of the tumor volume.

This model actually causes several issues. First an issue of parameter identifiability: indeed, b and d can be easily compensated, and therefore none of them can be found to be identifiable. For that reason, and since the only element that matters is not the exact value, but the proportion between the different parameters in this equation, we decide to set b as 1. This way, we manage to obtain a model for which parameters are identifiable.

A second issue is a stability one when we want to optimize the different parameters. Indeed, for an important range of starting points, the algorithm will converge very slowly. The starting points for parameters in our algorithm therefore need to be chosen very carefully.

4.3.7 Limits

If the models that were previously defined seem totally consistent theoretically, they are faced with some difficulty when confronted with actual data. Indeed, as it can be noted in figures 18 to 23, the models do not seem able to fit the data properly, in particular the early phases. Notably, the whole pre-treatment and growth arrest phases described on figure 15 can not be fitted efficiently. This is a true problem, for it necessarily reduces the relevance of all the models to fit this kind of data.

This observed result is actually rather unsurprising, for the effect of the treatment is in these models only considered by the drug concentration in the organism. The more drug concentration is in the organism, the more tumor growth should be reduced, according to these models, which, as we saw in the 'Shape of the treated data' subsection, does not seem to be the case. Therefore, no model is able to predict a change in the kinetic response when the treatment does not change ; yet, it is the case in the data we are working on. Besides, a good part of these models include parameters that are not identifiable when confronted with actual data.

Since no model in the studied literature seems to be able to model these various phases, it appears useful to create a new model that will be better at evaluating the primary tumor kinetic response to treatment.

4.4 SunitinibTGI model

4.4.1 Required features

Before starting to actually build the model, it is important to detail the features that our model will have to contain to be deemed satisfying.

- Ability to perform a better fit on the early stages of the treatment:

this is precisely the reason we build this new model for.

- Better overall fit quality: we want this new model to bring overall better statistical results on the fit of all phases, compared with models in the literature.

- Simplicity of the model: we want the model to contain mathematical expressions as simple as possible, so that the calculation time to find a solution to the model is as short as possible.

- Biological meaning of the parameters: we want this model to include parameters conceived so that their values may bring us biological information on the situation and the problem we are working on.

- Parameter identifiability: we want this model to have parameters being identifiable. We set a tolerance maximum threshold of 50 % on the error (NSE) as our limit for parameter identifiability.

4.4.2 Shape of the selected model

The model is built upon 3 phases: during the first phase, 'pre-treatment' the growth is controlled by two parameters (a and γ_1). When the treatment starts and is efficient, a second phase 'growth arrest' begins ; the growth is controlled by 2 parameters ($\bar{a}$, γ_1) , and the duration of this phase is controlled by a parameter τ. Then, when the tumor regrows and the efficacy of the treatment seems to gradually fade, a new phase, called 'resistance phase' starts, controlled itself by two parameters (a, γ_2).

We can note the differences in growth kinetics between the three types of data that we had distinguished in the part 'Shape of the treated data':

- For the data sets that follow 'behavior breast 1', we can see that the three phases are represented and that the parameter $\bar{a}$ can be chosen equal to 0.

- For the data sets that follow 'behavior breast 2', we can note that only the first two phases are represented (hence, no need to consider the γ_2 parameter), and that $\bar{a}$ is there, also, equal to 0.

- For the data sets that follow 'behavior melanoma', we can note that only the last two phases are represented, and that $\bar{a}$, in this case, cannot be picked as equal to 0.

4.4.3 Expression of the model SunitinibTGI

The conceived model, a 5-parameter model, can be written this way:

$$\frac{dV}{dt} = \begin{cases} aV^{\gamma_1} \text{ if } t \leq t_{start} \\ \bar{a}V^{\gamma_1} \text{ if } t_{start} < t \leq t_{start} + \tau \\ aV^{\gamma_2} \text{ if } t > t_{start} + \tau \end{cases} \tag{12}$$

where t_{start} is the day at which the treatment starts to be administered.

The choice to use three successive Power Law models can be justified by the analysis performed in the part 'Comparison of the accuracy of the untreated tumor growth models', which showed that this model was satisfying to fit untreated data, and that it had biological relevance.

Besides, it is a simple model (an explicit solution can be very easily found for each one of the 3 phases: $V(t) = (V_0^{1-\gamma_1} + a(1-\gamma_1)\ t)^{\frac{1}{1-\gamma_1}}$), with parameters that have a biological meaning, relevant to the context.

A biological interpretation possible for this model is the following.

Of course, when no treatment is administered, the situation is exactly the same as for the control groups. When the treatment starts and is still efficient (second phase), we suppose that it prevents the cells from proliferating and will modify the value of the proliferating rate a into a $\bar{a}$. When the treatment loses a part of its efficacy (third phase), we suppose that it is related to the fact that the tumor cells have developed a form of resistance to sunitinib. This resistance may be biologically related to the opening of a another way to pursue angiogenesis. Among the hypotheses that we could draw is that when the antiangiogenic drug is administered at first, it completely blocks and starves the tumor cells that would otherwise grow through the receptors (RTKs) blocked by sunitinib. But not all cells grow thanks to these receptors. Some of them may grow through other cellular pathways. Some may notably grow through hypoxia, and do not require the same amount of nutrients. This way, the angiogenesis would start over when those hypoxic cells would get more significant in the whole pool of tumor cells. Unaffected by the antiangiogenic drug, they would be able to grow in spite of the administered treatment. But, since their vasculature is much less stable, their growth would be less strong than tumor cells that would

grow through RTKs. Anyway, this change in vasculature could help us justify this switch of γ between the two phases. Of course, this interpretation can not be confirmed, but is one way of seeing the model with biological relevance.

Still, on a mathematical point of view, two major problems seem to rise with that definition of the model.

4.4.4 Smoothing of the model

First, the function that is calculated is not continuously differentiable, for its derivative is of course discontinuous.

This causes a problem, notably related to parameter identifiability. Indeed, statistical analyses that were performed showed that the lack in differentiability for the model led to significantly larger NSEs for all parameters, preventing parameter identifiability in all cases.

For that reason, we decided to smooth the model to make it continuously differentiable, and in particular the explicit solution, since a statistical analysis that was performed showed that it was the smoothing that led to the best improvement in stability. With this new smoothing, the model writes as:

$$V_1 + (V_\tau - V_1)RT_d + (V_2 - V_\tau)R_\tau \tag{13}$$

where V_1 is the explicit solution of the model in the first phase, V_τ the explicit solution of the model in the second phase and V_2 the explicit solution of the model in the third phase. As for the smoothing terms, they can be expressed as:

$RT_d = 0.5(1 + \tanh(p_{cont}(t - t_{start}))$

$R_\tau = 0.5(1 + \tanh(p_{cont}(t - t_{start} - \tau))$

In this smoothing term, p_{cont} is chosen by trial and error (here, its value is set to 2.1). The logic behind the use of the smoothing term RT_d is that, when $t << t_{start}$, $RT_d \approx 0$. When t draws near t_{start}, RT_d draws near 1, and it remains close to 1 $\forall t \geq t_{start}$. Thanks to this, we thus obtain a continuously differentiable model. This mathematical process is only there for parameter identifiability, and so that it checks mathematical properties that will make it easier to be used.

Model	Par.	Unit	Median value (CV)	NSE (%) (CV)
SunitinibTGI	γ_1	-	0.561 (6.89)	3.02 (28.2)
	τ	-	3.27 (124)	50.1 (4.69e+03)
	γ_2	-	0.398 (3.69)	2.04 (11.5)

Model	Par.	Unit	Median value (CV)	NSE (%) (CV)
SunitinibTGI	γ_1	-	0.518 (5.7)	1.18 (61)

Model	Par.	Unit	Median value (CV)	NSE (%) (CV)
SunitinibTGI	γ_1	-	0.648 (2.75)	2.4 (40.6)
	γ_2	-	0.596 (4.68)	3.46 (43.3)

Figure 24: *Parameter identifiability for the SunitinibTGI model [the first table shows the maximal uncertainty for τ]*

4.4.5 Parameter identifiability in the model SunitinibTGI

The second problem is the question of parameter identifiability. Indeed, since the sunitinibTGI model is a 5-parameter model, it is unlikely that all parameters can be found to be identifiable when confronted with the data. But, fortunately, we performed analyses that showed that a certain amount of parameters could be set, without damaging fit quality.

First, by performing the likelihood ratio test between a model with a set a and a free a, we see that we can set a with a very thin loss in fitting quality. Besides, this a can be set at the same value for all cells belonging to the same cell line. It is therefore consistent (and biologically relevant) to consider a as the average of the a obtained from the fit of the control groups with a Power Law model. Indeed, since a is supposed to model growth before any treatment is even administered, it is consistent that it could be learnt from control data. Setting a leads to a first improvement in parameter identifiability.

Yet, the same statistical analyses (likelihood ratio test) showed that it

was not interesting to set γ_1 to the average of the control groups γ fitted with a Power Law model, for it led to a dramatic decrease in fit quality. This result can be explained by individual variability between mice, and because the influence of γ_1 on the trajectory of our graph is much more important. For that reason, individual variability can really make major differences. It is possible however to set an a priori on γ_1, to pick it as a variable following the γ control distribution fitted with Power Law model. This makes sense biologically, for the same reason as the fact of setting a ; these parameters control the pre-treatment tumor growth, and it is thus relevant to consider that they can be learnt from the control groups. There again, restricting the range in which we can choose γ_1 leads to an improvement in parameter identifiability.

The third step towards an even better parameter identifiability is the setting of $\bar{a}$. This decision is not really motivated by a biological reasoning, but rather by data analysis, and statistical analyses analogous to the ones who led to the setting of a and the restrictions on γ_1. They showed that $\bar{a}$ could be set at the same value for all cells from the same cell line. $\bar{a}$ controlling tumor growth in the early stages of the treatment administration, it would not be consistent at all to learn it from control groups. For breast cancer cells, since we have a clear arrest in tumor growth during this phase, we quite simply choose $\bar{a}$ equal to 0. For melanoma cells, we first fit a model with a free $\bar{a}$. The median value of $\bar{a}$ obtained by this process is then the value to which we will set $\bar{a}$ in ulterior analyses.

This way, we end up obtaining a model with only 2 parameters left totally free (τ and γ_2), and one parameter constrained in a specific distribution (γ_1). We therefore improve quite significantly the parameter identifiability of the model [see figure 24].

4.4.6 Comparison with classical models

This model was compared, for each data set, to the other classical models for treated tumor growth ; the results are displayed on figure 25. We can note that for all data sets, and for all statistical tools, the SunitibTGI models seems better fitted to model the process of treated tumor growth.

4.4.7 Notable results related to the model ([14])

Now that the model has been shown to be both efficient in terms of fitting capacity and in terms of parameter identifiability, we can try and evaluate the values of its parameters on the various data sets, and hence note a few interesting elements on the biology of the problem.

Goodness of fit

Model	SSE	AIC	RMSE	R2	p > 0.05	#
SunitinibTGI	**0.115(0.0554 - 0.151)[1]**	**-45.3(-61.5 - -37.3)[1]**	**0.339(0.238 - 0.397)[1]**	**0.955(0.931 - 0.982)[1]**	100	3
Simeoni2004	0.198(0.132 - 0.348)[2]	-30.6(-38.5 - -15.3)[3]	0.441(0.368 - 0.596)[2]	0.937(0.865 - 0.969)[2]	100	4
Simeoni AA free growth	0.22(0.132 - 0.348)[3]	-30.6(-40.5 - -17.3)[2]	0.462(0.368 - 0.596)[3]	0.935(0.865 - 0.969)[4]	100	3
GompertzNS	0.395(0.277 - 0.58)[4]	-15(-22.8 - -5.07)[4]	0.629(0.531 - 0.765)[4]	0.931(0.905 - 0.958)[5]	100	3
Hahnfeldt	0.551(0.451 - 0.632)[5]	-6.45(-11.1 - -3.01)[5]	0.744(0.672 - 0.803)[5]	0.908(0.869 - 0.933)[6]	60	3
Exp-log kill	0.704(0.523 - 0.842)[6]	-2.88(-9.55 - 1.87)[6]	0.836(0.724 - 0.918)[6]	0.937(0.921 - 0.952)[3]	60	2
DynaCC	0.751(0.57 - 0.883)[7]	0.691(-5.51 - 5)[7]	0.865(0.757 - 0.945)[7]	0.906(0.887 - 0.933)[7]	60	3

Figure 25: *Comparison of the SunitinibTGI model with classical models*

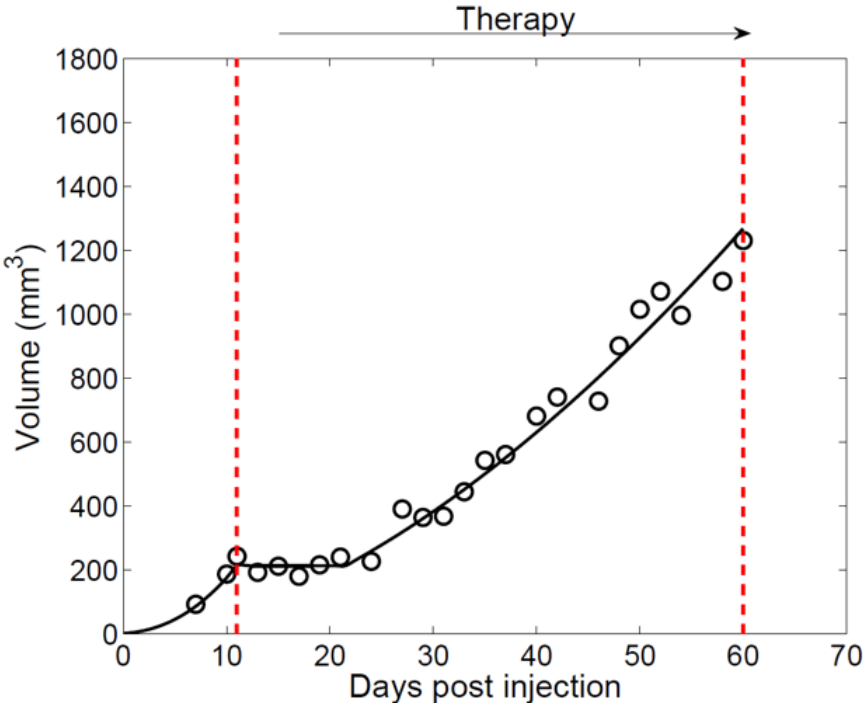


Figure 26: *SunitinibTGI modeling for an individual mouse injected with breast cancer*

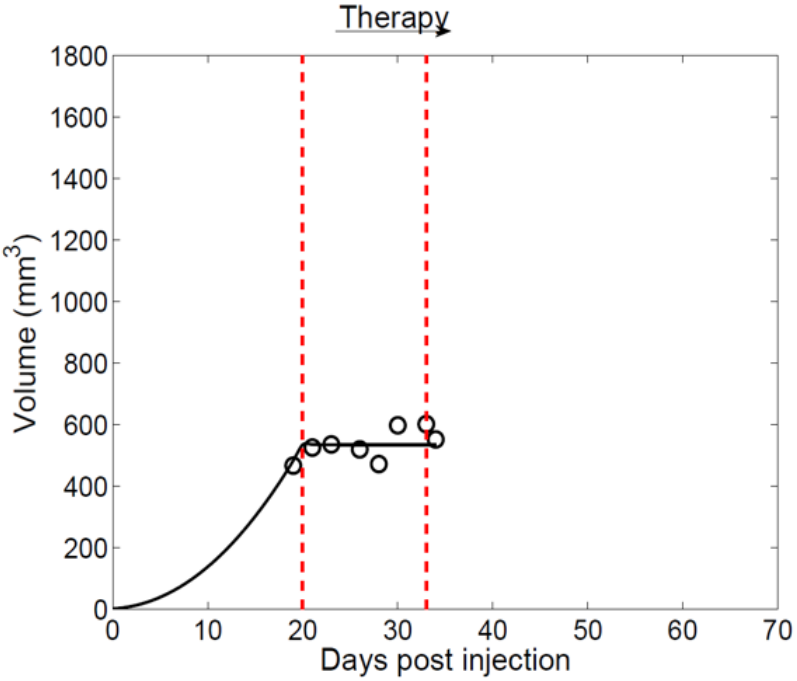


Figure 27: *SunitinibTGI modeling for an individual mouse injected with breast cancer*

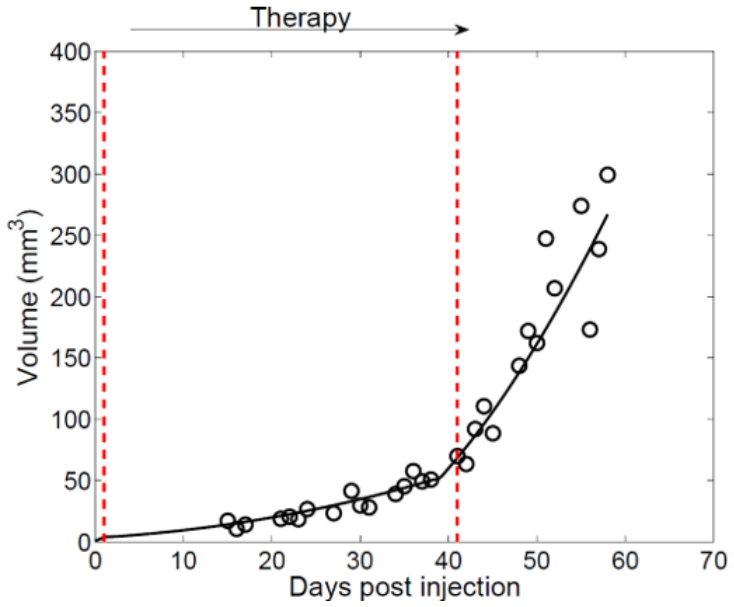


Figure 28: *SunitinibTGI modeling for an individual mouse injected with melanoma*

1. We notice that during the 'resistance' part of the treatment for breast cancer, tumor growth is significantly weaker than growth without a treatment at all. Indeed, if we perform for instance the analysis on groups 31, we notice that we obtain medians $\gamma_1 = 0.56$ (for a NSE of 3%) and $\gamma_2 = 0.40$ (for a NSE of 2%).

   It seems to imply that even after the treatment has its efficacy clearly reduced, it still has an impact, and a noteworthy influence to reduce tumor growth. The interpretation based on hypoxia that was described in the 'Expression of the model SunitinibTGI' subsection could be an explanation of it.

2. We do not have a sufficient amount of data to be able to model and compute the optimal drug scheduling, as we would want to. Indeed, since a scheduling rests on two significant elements (dose and administration time), to be able to perform a proper analysis, we would need to have examples with one of those two features (only) as common elements, and the data sets that were provided, did not allow us to obtain this kind of result. Still, the parameters that are computed by the model can help us make a decision between 2 different scheduling.

   For instance, we can note that for group 31, which has the following scheduling (60 mg/kg on 40 days), we have medians $\gamma_1 = 0.56$ ($NSE = 2\%$) and $\gamma_2 = 0.40$ ($NSE = 3\%$) ; for group 32, with the following scheduling (120 mg/kg on 7 days), we have medians $\gamma_1 = 0.57$ and $\gamma_2 = 0.52$ ($NSE = 2\%$).

   We may note that a larger dose in a shorter time period, leads to a ultimate tumor growth much higher later, and to a similar tumor growth during treatment. Therefore, if we are interested in reducing as much as possible the tumor size before surgery, for instance, we can see that we should, generally, privilege treatment 31.

3. For the melanoma data, and the sunitinib-resistant cells, we first notice that is is statistically, and in terms of fitting capacity, advantageous to set τ at the value for the time of end of the treatment. We can conclude from it that in this specific situation (melanoma/sunitinib-resistant cells), the end of the efficacy phase coincides exactly with the end of the treatment. We then deduce that in the case of melanoma, there is no post-treatment efficacy phase ; once the treatment is stopped,

tumor automatically regrows without any delay. And, more importantly, the treatment does not look as if its efficacy were significantly reduced during its whole administration.

4. We also note that the higher the tumor volume is at the beginning of the treatment, the higher τ will be. Indeed, for the group 31 for instance, which has an initial tumor volume of 196.7 mm^3 we have $\tau = 5.28$ days ($NSE = 39\%$) ; and for group 51 (initial tumor volume: 439.5 mm^3), we have $\tau \geq 14$.

   Hence, the higher the initial tumor volume is, the longer the growth arrest phase will last. This result tends to give credit to the hypoxia interpretation of our model that we had made in the subsection 'Expression of the model SunitinibTGI': the higher the number of cells that would grow with the RTK pathway is, the longer the antiangiogenic drug will have to proceed to completely eliminate those cells to give place to the hypoxic ones (related to treatment resistance).

4.4.8 Limits of the model

This model is not exempt from problems and limits. Among them, we can note that no direct and explicit connection to pharmacokinetics has been clearly established. The impact of pharmacokinetics is obviously in the parameters, but the exact relationship that connects, for instance, drug concentration and the value of some of the parameters, is not obvious at all.

One idea that was developed during this internship to connect them turned out to be fruitless.

Considering the formula $\tau = \inf\{t, \int D(t) \geq s\}$ (where D is the administered dose, and s a parameter to be optimized) was meaningful, for it implied that above a certain threshold of concentration, the drug partly lost its efficiency, and the tumor cold grow again. Unfortunately, as it can be evidenced with the fact that τ depends on the initial tumor volume, obviously completely unrelated to the administered dose, this was not a correct way of seeing the problem.

Establishing an explicit connection with pharmacokinetics would bring a supplementary layer to the model, but during this internship, it did not look that obvious to make.

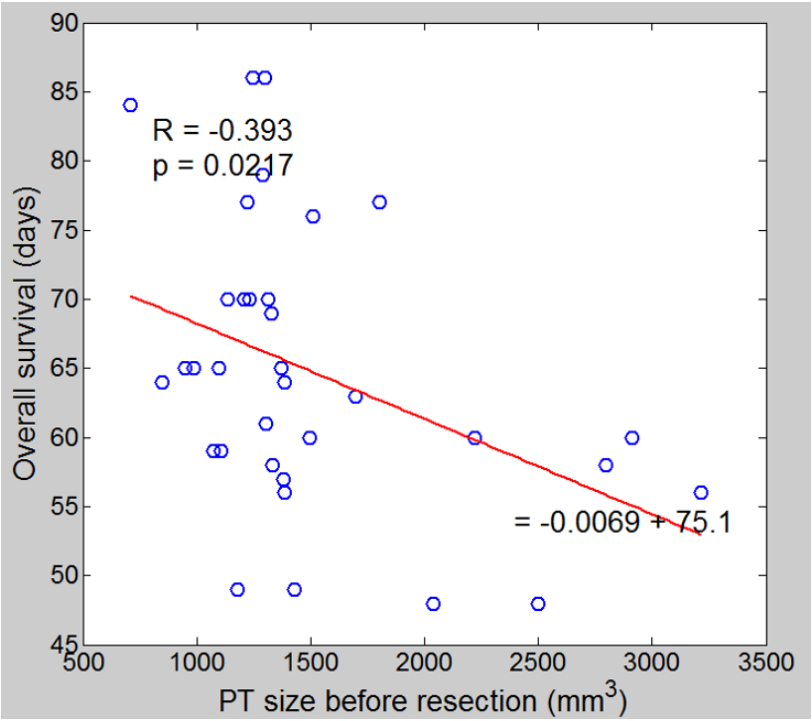


Figure 29: *Correlations overall survival - tumor size pre-resection for control groups*

5 Data correlations

Correlations between various data sets were also calculated. 2 of them actually stood out from the rest.

The first correlation is between overall survival and the size of the tumor prior to resection. As shown in figures 29 and 30, we note that for control groups, the lower the primary tumor size is before resection, the longer the mouse can be expected to live, which makes sense from a biological point of view. But, for mice that are treated, this correlation does not exist any more (and even an inverse correlation seems to be existing, even though not statistically significant). This is quite surprising. Among the possible explanations are that some tumors are more affected in their size reduction than others by sunitinib, making the correlation that was spotted in control groups fade away. This interpretation would give a neutral role to sunitinib in terms of evolution of overall survival between control groups and treated ones. Another possible explanation would be that when sunitinib acts the most (tumor size reduced before resection), the overall survival would

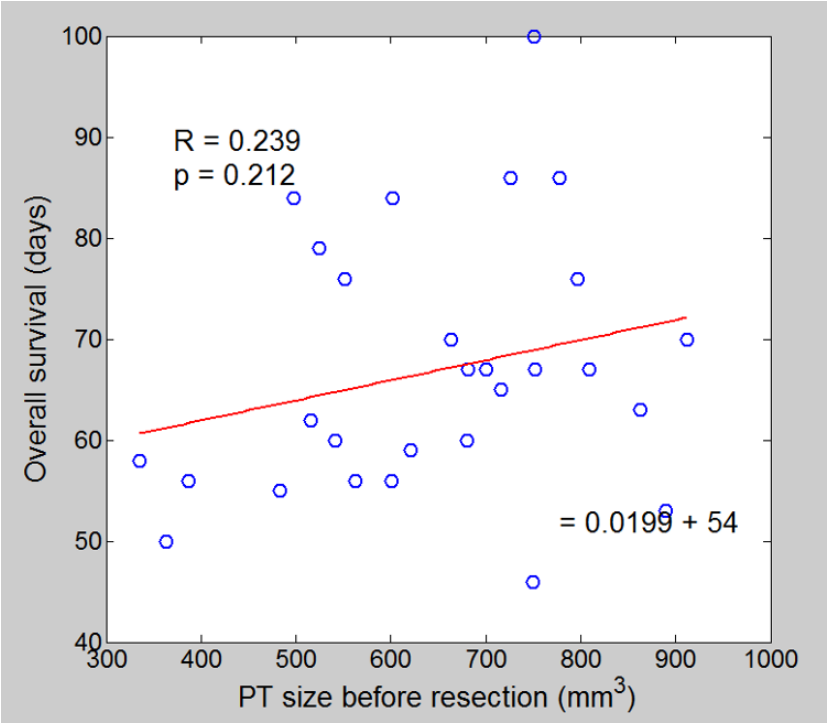


Figure 30: *Correlations overall survival - tumor size pre-resection for treated groups*

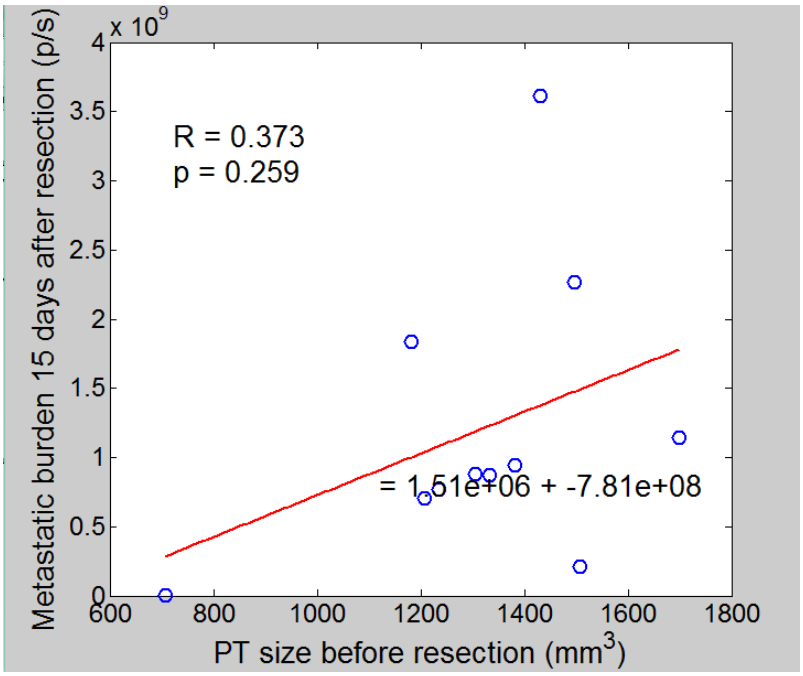


Figure 31: *Correlations tumor size pre-resection - metastatic burden for control groups*

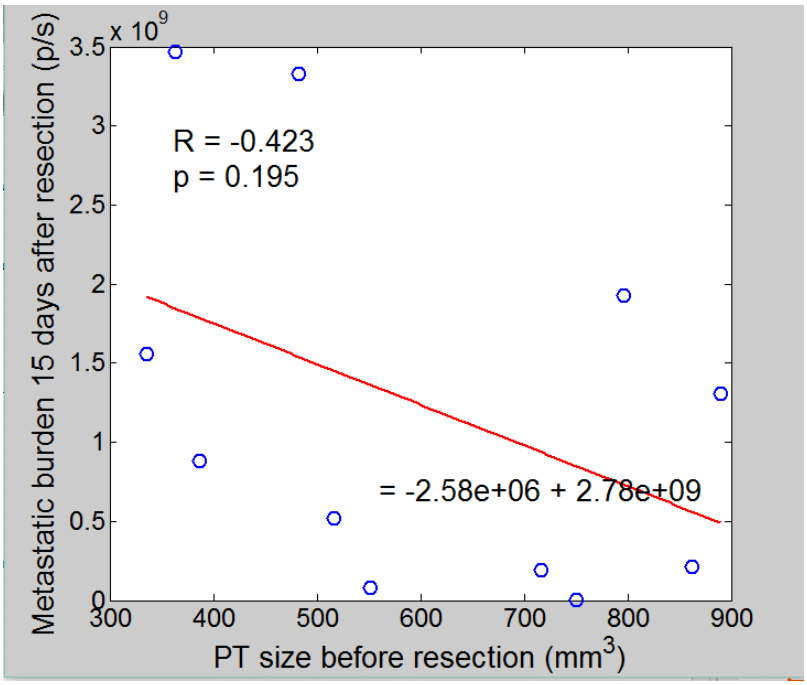


Figure 32: *Correlations tumor size pre-resection - metastatic burden for treated groups*

tend to decrease, which would give a harmful role to sunitinib. From this analysis only, we cannot choose a perfect interpretation, but it is still an interesting result to consider.

The second correlation which is interesting to consider (figures 31 and 32) is between the size of the primary tumor before resection, and the metastatic burden 15 days after resection. What we can note is that for the control groups, the higher the primary tumor size is, the higher the metastatic burden will be, which, once more, makes sense biologically. But for mice that were treated with sunitinib, the correlation does not exist anymore (and even seems to be, even though statistically insignificant, inverse). This can also be explained with an asymmetrical influence of sunitinib: it would affect some tumors more than others all while not modifying the related metastatic burden, its role would therefore be neutral in the issue. Or we can see it as, when the tumor size is lower, then the sunitinib has acted more, and a more significant metastatic burden can be found, which would, under those circumstances, show that the sunitinib would indeed provoke more metastatic burden. It would then be harmful. But once more, from those singular experiments, we cannot know which interpretation is sound.

6 Computing the metastatic burden ([7], [18])

Our model for primary tumor growth SunitinibTGI can then be used to try and describe the metastatic burden, as evidenced by the following section.

6.1 General equation

Based on [7], we see that the evolution of the metastatic burden can be evaluated thanks to the following transport equation:

$$\begin{cases} \frac{\partial \rho}{\partial t} + \frac{\partial(\rho g)}{\partial V} &= 0 \\ g(V_0)\rho(t, V_0) &= \mu V_p(t) \\ \rho(0, V) &= 0 \end{cases}$$

where $\rho(V, t)$ is the metastatic density that rules the metastatic population, for a given volume V at a specific time t ; g is the function that describes the growth ; $V_p(t)$ describes the volume of the primary tumor ; μ is a parameter that describes the metastatic process (colonization

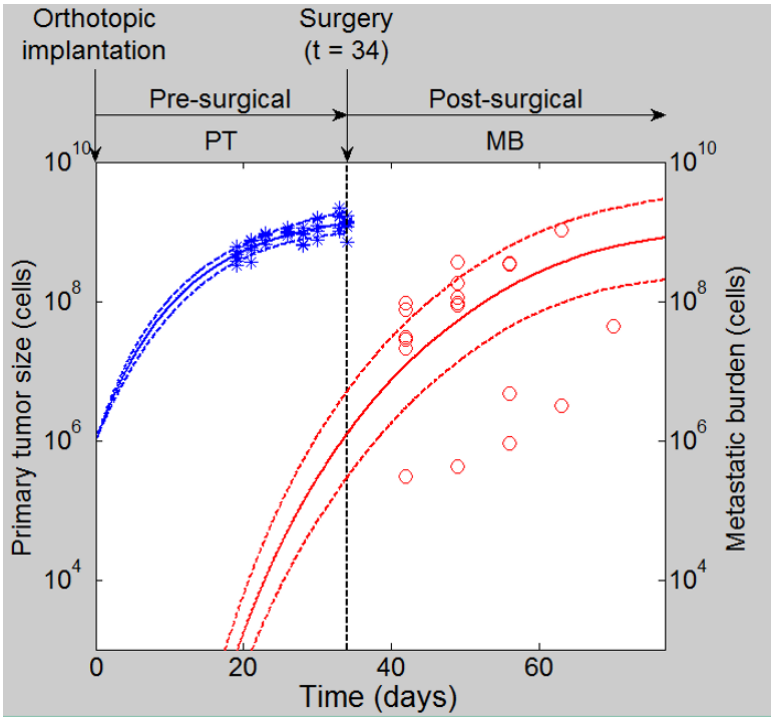


Figure 33: *Evolution of the metastatic burden (control groups)*

and dissemination).

Thanks to the analyses that were led in the previous part, we know that we have obtained a model allowing to evaluate the volume of the primary tumor $V_p(t)$ more precisely than classical models. Here, we then choose $V_p(t)$ following the SunitinibTGI model, g following a Gomp-Exp model, and we approximate this equation numerically using the method of characteristics.

6.2 Conclusions

When numerically solved with the method of characteristics, this equation returns us a ρ, which can be displayed in the graphs on figures 33 and 34. What we can note is that no clear acceleration is actually found computationally between the cases where no treatment is administered, and the cases where sunitinib is actually administered. This tends to show that the results of metastatic acceleration do not look like they actually have a total relevance on a statistical point of view.

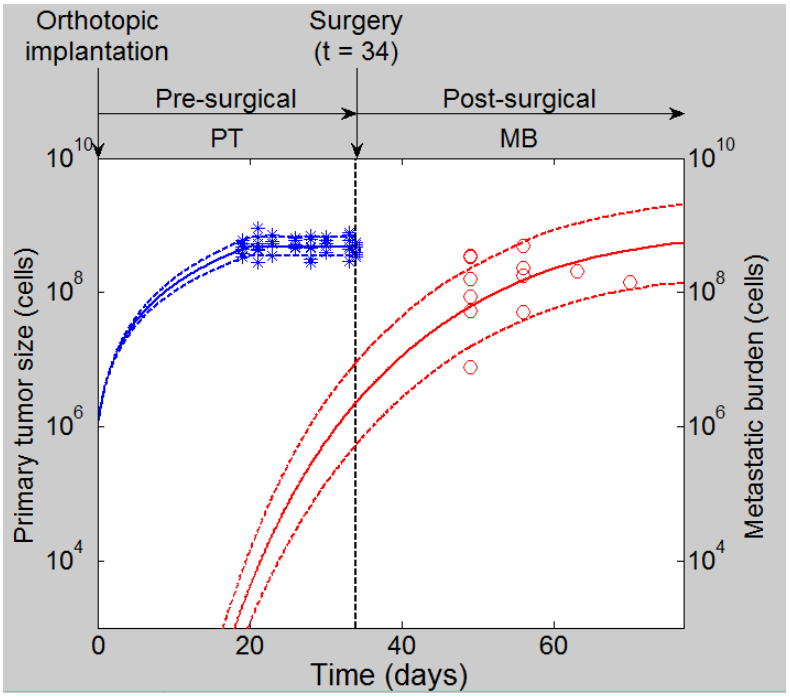


Figure 34: *Evolution of the metastatic burden (treated groups)*

6.3 Prospects

Some work still needs to be done in order to improve the metastatic process modeling. For instance, being able to distinguish (through an analysis of μ) the dissemination and the colonization in the metastatic process could lead us to more insightful results. Indeed, if we were able to distinguish these two parts of the metastatic process, we would therefore be able to see how each one of those two phenomena can be affected with external parameters independently, instead of considering the whole process as one.

In particular, in this case, to be able to model the impact of hypoxia, it could prove to be interesting to try and express μ as a function of $\frac{K}{V}$, where K is the vascularization and V the volume. This way, a possible connection between hypoxia and metastatic acceleration could be shown (or dismissed).

Finally, we have made sure all along that all of our parameters were statistically significant (low NSE,...). This also gives us hope that it may be possible to build, from that basis, a predictive model, which would be able, from a given number of input points for primary tumor

size and metastatic burden, to predict with a relative accuracy the evolution of the disease.

7 Conclusion

As a summary, we could say that the mathematical data analysis of the primary tumor and the metastatic burden evolution for mice does not show a clear connection between sunitinib administration and metastatic acceleration.

This result was obtained after an optimization algorithm benchmark showed that the Nelder-Mead or the Levenberg-Marquardt algorithms were the most fitted to deal with the problem. Among the classical models of untreated tumor growth, the Power Law seemed to be one of the most interesting in terms of statistical results and biological meaning. When working on treated tumor data, we saw that no classical model was really able to describe accurately the evolution of the primary tumor. For that reason, a new model, whose properties were checked, was built. This new, more efficient model, allowed to find new interesting results on the evolution and the primary tumor, and on correlations between some elements of data. Incorporating this model into a classical metastatic equation helped show that the acceleration was clearly not obvious from a mathematical point of view.

8 Annex

8.1 Cancer biology ([9])

8.1.1 Overview

Cancer is, in France, the first cause of mortality: about 30% of people die from this disease. In spite of the intensive research that is being carried out, cancer remains a disease that is not entirely understood. The aim of this annex is to describe, as a complement to the rest of the internship report, the basements of cancer biology.

To describe it in a few words, cancer is a disease which induces the uncontrolled growth of a group of cells, the primary tumor, within an organ. At a certain stage, some cells from the primary tumor manage to leave the original organ through the blood vessels or the lymph nodes of the organ, to settle in a new organ. The cells that carry out this trip through the body are called metastases. Metastases are, in most cancers, the main cause of death, and they also explain the difficulty in treating cancers ; fighting the primary tumor is not enough to cure the patient, since other tumors may grow in other parts of the body.

This annex deals with two different subjects: a description of the disease itself and of its biological mechanisms, and a little summary of the possible treatments.

8.1.2 Detailed description of the disease

During a cancer, some cells will grow and divide abnormally. The new structure that then rises from this uncontrolled growth is called a tumor. But for the tumor to actually trigger a cancer, it must satisfy a certain range of conditions, such as:

- the ability to sustain proliferative signals. The cancer cells are not controlled in their proliferation by the environment, as opposed to normal cells in organism.
- the ability to evade growth suppressors. The cancer cells have the ability to ignore the signals that restrict cell growth and proliferation.
- the ability to resist cell death, in particular apoptosis. Apoptosis is the programmed cell death, which notably occurs when the cell

is abnormal. The deactivation of the signal can then prevent the organism from destroying cancer cells.

- the ability to get replicative immortality. Unlike all normal cells, cancer cells can process an unlimited number of cell cycles, and undergo mitosis an unlimited number of times.
- the ability to induce angiogenesis: the primary tumor can send itself signals to create blood vessels, which will supply the tumor in nutrients, and will allow it to grow even more.
- the ability to invade the full body (with metastases) through blood vessels and the lymph system.

All these properties, common to all cancers, are the core ideas of what the disease is capable of, and of how it proceeds. In the following sections, we will focus more precisely on the evolution of the disease.

Genetics

Like all biological phenomena, cancer is a disease in which genetics plays an important part. Indeed, cancer is born from mutations which affect specific genes, and which lead to malfunctions in various phases of the cell cycle. The three main genes responsible for it are:

Oncogenes: oncogenes are mutated genes, which send signals to cause uncontrolled cell growth. The said cells are then liberated from the constraint of limited cell proliferation. If an oncogene appears after a gene mutation, the first step to cancer is reached.

Tumor suppressor genes: even once the oncogenes have appeared, defense mechanisms against cancer still exist in the organism. Among these mechanisms, we can count tumor suppressor genes. These are genes which have the ability to slow down (or even arrest) the cell cycle. This way, even if a cell proliferates too much because of oncogenes, the tumor suppressor genes still retain control on how it divides, and can stop the process or even induce the apoptosis (programmed cell death) of the cell. But if a mutation occurs, reducing (or erasing) the power of the gene, then the first defense mechanism of the organism is neutralized. In most cancers, the mutation of these genes occurs in the early stages of the disease.

DNA repair genes: another defense mechanism in the organism. These genes code for proteins which will solve the errors due to a mutation.

Therefore, they widely reduce the development of tumor cells. But they only reduce it, because these genes have limited repairing power, and if the error rate is considerably larger than the repairing capacity of the gene, the cells will still uncontrollably grow. And of course, a mutation of these genes, if it leads to a reduced power or a vanished power, greatly accelerates the development of the tumor cell.

If those conditions are verified, we have then reached the first stage of cancer. Due to these genetic mutations, the cells will grow uncontrollably in the organ, and give birth to a large cell structure, the tumor. Yet, at this point, the tumor can still be benign, and can fail to develop more. For it to happen, several events need to occur.

Development of the tumor inside the organ

The first thing that comes to mind is the fact that to grow, the tumor needs to be supplied in nutrients through the blood. In some cases, the tumor cells, perceived as abnormal by the organism, are not supplied. This way, the tumor cells cannot live long, and are destroyed not long after they are created. Thus, even if the tumor cells undergo uncontrollable growth, since they can not stay alive, it is not an immediate problem.

The issue is raised when the tumor is supplied in blood and nutrients. This is related to the fact that the tumor, in most situations, is not made of cells which are all abnormal. Indeed, the tumor often manages to include normal cells in the structure (the normal cell tissue within a tumor is called the stroma). This way, the organism, recognizing these cells as normal ones, will supply the tumor in nutrients and blood. The tumor will then be able to grow, while being fed by the organism itself.

It is good to note that at this stage, the tumor can still be considered as benign. Therefore, it is not -yet- really dangerous for the patient, unless the tumor is located in a very fragile and specific organ (such as the brain for instance). If the illness is treated at this step, it can be cured with no problem.

The tumor has then managed to grow to a much larger size (generally around $1mm^3$). But it is still limited in its development by the amount of blood that is supplied; as it is now, it is still not a danger for the individual. For it to grow more, the tumor has to receive more nutrients. The tumor, in all cancers, ends up being able to induce a phenomenon called angiogenesis.

Angiogenesis is a physiological process through which new blood vessels are created. It is a normal and vital process (crucial for wound healing for instance), but it may be impeded by the tumor so that it can grow more. In all cancers, the tumor will be able to induce angiogenesis, leading to the development of a vascular system within the tumor. This way, the tumor will be supplied in more nutrients, and will be able to grow even more and to considerably develop. Once this step has been reached, the likelihood of cancer development considerably rises and the illness needs to be treated as quickly as possible.

On a biological level, angiogenesis in induced by the tumor through the VEGF protein (Vascular Endothelial Growth Factor). This protein is part of the system that restores the supply in oxygen when blood circulation is inadequate. Signals leading to the production of VEGF proteins are emitted by the tumor itself. The tumor, through previous mutations, carries VEGF-R, receptors for the VEGF proteins. This way, the tumor stimulates the production of VEGF proteins for itself, which makes the local vascular system expand. The tumor, being supplied in more nutrients, can therefore grow again. It is noteworthy that all the receptors related to this type of protein signals can be called receptor tyrosine kinases (RTKs).

We have then reached the second stage of cancer.

Metastatic process

The tumor is now a very large cell structure located inside an organ, and which is able to grow because of angiogenesis. Yet, the cancer is not (unless it is located near the brain) a deadly threat for now on. It will become one if it is capable to metastasize.

Metastatic dissemination is the process by which some cells from the tumor will leave the organ in which they are located to reach a new organ (through the blood vessels or the lymphatic system), where they will settle and create a new tumor. This way, the cancer can disseminate in the entire organism. This metastatic process is responsible for 90 % of patients' death. For this reason, many studies are carried out to try and understand te phenomenon.

Epithelial to Mesenchymal Transition

The EMT (Epithelial to Mesenchymal Transition) is a recent discovery, which may explain the process by which cells from the tumor can leave it to join a blood vessel or a lymph node.

Epithelium is a type of organic tissue, which lines the surface of blood vessels and organs. It is made of a large number of epithelial cells bound together. The epithelial cells are not free to move, they are all connected with very tight junctions.

On the other hand, mesenchyme is a type of tissue which is characterized by associated cells (mesenchymal cells), loosely bound together. The mesenchymal cells can move rather freely and are not tightly connected to one another.

It has recently been shown that some cancerous cells, mostly located in the periphery of the tumor, were able to move from an epithelial cell state to a mesenchymal one. Thanks to this transition, the cells are able to move more freely, and they can then join blood vessels or the lymph nodes, in order to reach another organ. They become metastases. Once this step has been reached, we are at the third stage of cancer.

Dissemination and colonization

Once in the blood vessels, the metastases have to escape the immune system (few cells will actually survive), which often perceives them as stranger cells, and then destroys them. What ensues can be divided into two parts: dissemination and colonization.

During the dissemination phase, the metastases will be in the blood or the lymph nodes.

At some point, the metastases can reach a new organ and settle there, through MET (Mesenchymal to Epithelial Transition), the inverse of the EMT process. Thanks to this, the metastases can form a new structure (secondary tumor) where they will be jointly bound (as an epithelial tissue) in the new organ. They have thus colonized the organ, and the secondary tumor will start to grow again, and eventually to metastasize. Once an organ has been colonized, we have reached the fourth stage of cancer.

It can be interesting to note that the early stages of the colonization phase can be very long, and even seem like dormancy. Indeed, for a while, at the beginning of the colonization, the size of the tumors do not change, and the health of the patient does not worsen either.

But once this dormancy phase is over, metastases will grow more and more aggressive, and the health of the patient will just go declining.

8.1.3 Current treatments

Some treatments exist to improve the life and the health of ill patients. If they are presented separately in this part, they are very often combined for more efficacy.

Chirurgy

The first - and seemingly more obvious - treatment is the resection (surgical removal) of the primary tumor. It seems like a pretty intuitive step, but the effects of such an operation are contrasted. If for some patients, a notable improvement in health was noticed, it is not the case for all in general. Indeed, one interesting aspect concerning the primary tumor is that it both sends proliferative signals to itself, but that it also sends inhibition signals. Removing it has led, in some cases, to a notable acceleration of metastatic growth, and then a worsening of the state of the patient. Therefore, the issue of whether removing surgically the tumor is the best option, when it is not located in an organ where it is directly lethal, is still an open one.

Chemotherapy

The chemotherapy is the administration of the combination of several drugs, which will have as a common behaviour, to target the cells that divide very quickly. Of course, the idea is this way to destroy the tumor and metastatic cells, which divide uncontrollably. But not only cancerous cells are destroyed through this process: some blood cells, hair cells, among others, will also be targeted. For that reason, chemotherapy is a very important type of treatment, but with a lot of adverse effects.

Targeted therapies

The idea of targeted therapies is, instead of destroying the cells directly, to attack it through its biological mechanisms, through extracellular (outside the cell) or intracellular pathway. One of the most useful ways to fight cancer is to combat angiogenesis, with what is called antiangiogenic drugs. Let us consider the example of sunitinib.

The sunitinib is a molecule which inhibits cellular signaling by targeting receptor tyrosine kinases (RTKs), including receptors for vascular endothelial growth factor receptors (VEGF-Rs). The inhibition of these targets allow to reduce tumor vascularization, and can lead to the apoptosis of several cancer cells. This may allow to shrink the tumor.

8.2 Statistical tools

We display here a little overview of the various statistical tools used for data analysis in the report.

Let us consider a data set $(y_i)_{i\in[1,n]}$ and f the model considered to fit the data (with $(t_i)_{i\in[1,n]}$ the time range)

8.2.1 Sum of squared errors (SSE)

The sum of squared errors (SSE) can be defined as:

$$SSE = \sum_{i=1}^{n}(y_i - f(x_i))^2 \tag{14}$$

This statistical value is then simply the error, added for all points, committed by considering the model instead of the data. The best model, in that regard, is then the model for which the error is the smallest.

But the criterion, as helpful as it may be, is often insufficient to fully compare 2 different models.

8.2.2 Root mean squared error (RMSE)

The root mean squared error (RMSE) can be written as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - f(x_i))^2}{n}} \tag{15}$$

RMSE is a criterion which is very close to SSE.

8.2.3 Akaike Information Criterion (AIC)

Let L be the maximum value of the likelihood function for the model, and k its number of parameters. The AIC is then defined as:

$$AIC = 2k - 2\ln(L) \tag{16}$$

The model that has the lowest AIC is then the model that has to be favored. We notice that as the likelihood maximum is high, the AIC goes down, and that AIC grows with the number of parameters. This

criterion is then a way to choose the best model, all while penalizing the number of parameters. Indeed, the more parameters we add, the lowest the error will be ; but the number of parameters is also an important criterion to consider. This criterion takes it into account, by considering it as a penalty for a model.

This criterion is therefore adapted to the comparison of models with different number of parameters.

8.2.4 R^2 coefficient

Let us denote

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$

$$SS_{reg} = \sum_i (f(t_i) - \bar{y})^2$$

$$SS_{res} = \sum_i (y_i - f(t_i))^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{17}$$

The R^2 coefficient is then a measure of how much better it is to consider the model to fit the data instead of simply choosing the constant function, equal to the average of all data. R^2 can be negative ; in that case, it means that it is better to consider the constant function to fit the data. A negative R^2 is a sign that the model considered is not able to fit the data at all.

8.2.5 Log-likelihood ratio

The likelihood ratio test is a statistical test which allows to compare 2 models, when they do not have the same number of parameters. Let us call L_n the likelihood for the model with fewer parameters, L_a the likelihood for the model with more parameters. Then, the likelihood ratio is calculated this way:

$$D = -2\ln(\frac{L_n}{L_a}) \tag{18}$$

The likelihood ratio allows therefore to evaluate which one of the two models considered is to be favored.

8.3 Optimization methods for the Gompertz model

8.3.1 Introduction

Setting of the problem

As of today, the phenomenon of tumor growth is still an open problem. Indeed, no equation is considered as the equation which would perfectly model the data. Yet, for, notably, biological reasons, the Gompertz model is one of the most acclaimed models to describe this growth ([1], [2]). During this project, a simplified model of Gompertz growth, depending on 2 parameters, α and β, was considered. The goal of this project was to compute several optimization algorithms in order to find the values of α and β that allowed the best fit for the tumor growth data provided. This work is interesting for the results that were found about the efficiency of the optimization algorithms for a rather simple model can be extended to more complicated ones.

All the following results were computed using Python and its numerical (numpy) and graphical (matplotlib) libraries.

The Gompertz model

The Gompertz model is a model which is commonly used in the literature to describe tumor growth. It can be written as follows:

$$\begin{cases} \dfrac{dV}{dt} = (\alpha - \beta \ln(V))V \\ V(0) = V_0 \end{cases} \tag{19}$$

We notice that the form of the solution curve (figure 35) is compatible with a phenomenon of tumor growth. During the early stages of cancer, there are very few tumor cells, restricting, therefore, the growth rate. Once the number of tumor cells rises, the growth rate will undoubtedly increase until a certain stage. Indeed, since the resources for all cells are limited, the scarcity of the nutrients when there are too many tumor cells tends to gradually decrease the growth rate. This function seems to fit well the phenomenon of tumor growth.

In this project, we made the assumption that the tumor is $1mm^3$-long initially ($V_0 = 1$). Our work was to try and estimate the parameters

α and β which allowed the best fit for the data provided.

Data provided

The data that was provided to calibrate the model represent the growth of a tumor made from a lung cell line (LLC: Lewis Lung Carcinoma), subcutaneously injected into mice. The volumes were measured using a caliper. The largest (L) and smallest (w) diameters were measured subcutaneously using calipers and the formula $V = \frac{\pi}{6}w^2L$ was used to compute the volume (ellipsoid).

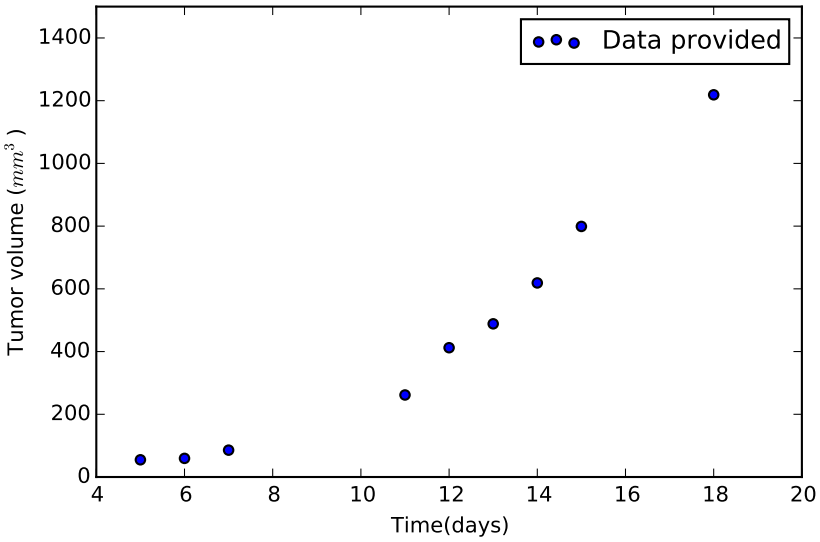


Figure 35: Data provided

8.3.2 Optimization problem

Optimization criterion

We want, in this project, to find the values of α and β that will lead us to a solution as close as possible to the data. For this reason, we choose our optimization criterion F as (with n the number of data provided, y_i the volume attained in t_i in the data, and M the exact solution to the Gompertz ODE):

$$F(\alpha, \beta) = \sum_{i=1}^{n} (y_i - M(t_i; \alpha, \beta))^2 \tag{20}$$

We will consider $\log(F)$ as the criterion instead of F for the gradient descent algorithms, since it makes convergence easier. We will then try to find the α and β minimizing the criterion.

Analytic solution to the Gompertz ODE

In this section, we calculate the function which solves the Gompertz ODE. By carrying out a variable separation method (function of the form $-\frac{1}{\beta}\frac{u'}{u}$), we end up obtaining the analytic solution, displayed on figure 36:

$$M(t; \alpha, \beta) = \exp(\frac{\alpha - \exp(-\beta t)(\alpha - \beta \ln(V_0))}{\beta}) \tag{21}$$

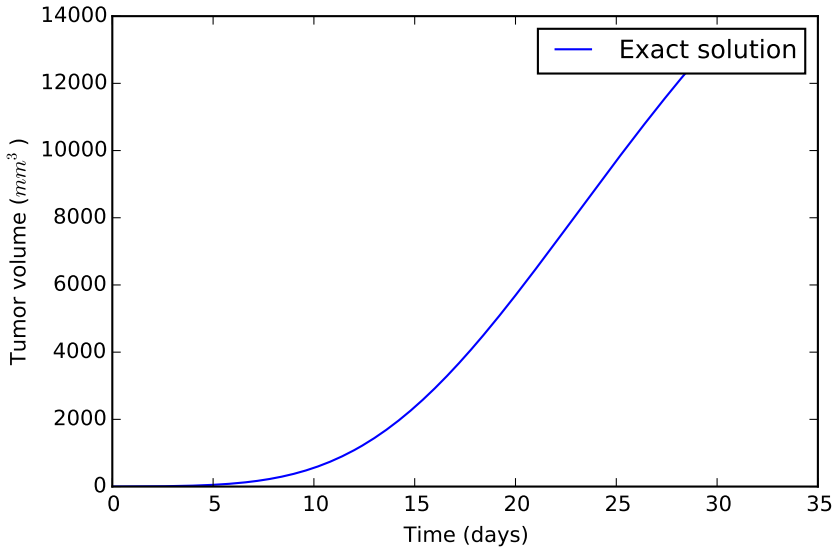


Figure 36: *Solution to the Gompertz differential equation, with* $V_0 = 1$, $\alpha = 1$, $\beta = 0.1$

Analysis of the optimization criterion

We plot on figure 37 the optimization criterion depending on the two parameters α and β. We can notice that there is a large number of local minima ("valley" of pseudo-minima), especially on figure 38, which will be a problem for some optimization algorithms that we will use afterwards.

This problem is clearly illustrated on figure 39, since we see that the contour lines of F depict very thin areas close to the global minimum, which may cause the algorithm to stop even if the point reached is not the one we wanted.

Besides, if we get a little further from the area with the more common values of α and β, we notice that the function is somewhat troublesome. Indeed, figure 40 shows that $\log(F)$ can take extreme values, which will make F impossible to evaluate numerically. This issue will be addressed in the part devoted to the Levenberg-Marquard algorithm.

8.3.3 Optimization algorithms

All the optimization algorithms which are studied in this report induce a propagation of the considered point in the descent direction of the optimization criterion. A descent direction is a direction along which the criterion has a strictly negative directional derivative; following it will then undoubtedly lead to a minimization of the function.

Gradient descent with constant step

Theory

The algorithm of gradient descent draws its logic from the fact that a function F will decrease the most in the direction that is opposed to the direction of the gradient. It takes for input an initial value x_0, which will be the starting point of the algorithm. The gradient descent algorithm generates a sequence of iterated points $x_1, x_2, ...$ and is stopped when a test is satisfied, sending as output the last point x_n generated. It follows 4 phases:

(a) Simulation: we first calculate $\nabla F(x_k)$.

(b) Test: we check if $||\nabla F(x_k)|| \leq \varepsilon$. If this is the case, then the algorithm stops, and the output x_k is returned.

(c) Calculation of the step: here, we consider a constant step α_k.

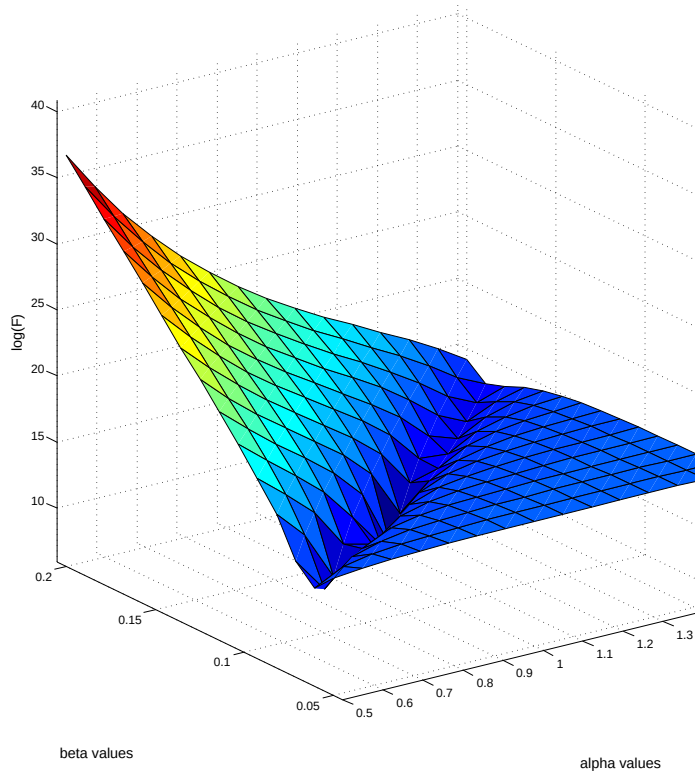


Figure 37: *F depending on α and β*

(d) New point: we consider a new point $x_{k+1} = x_k - \alpha_k \nabla F(x_k)$, and we start over to phase 1.

This way, we end up reaching a local extremum. But nothing, in this algorithm, can guarantee us than the value returned is a global minimum.

Analysis of the results

This algorithm was implemented ($\alpha_k = 0.01$, $\varepsilon = 0.0001$), and applied to the function $\log(F)$.

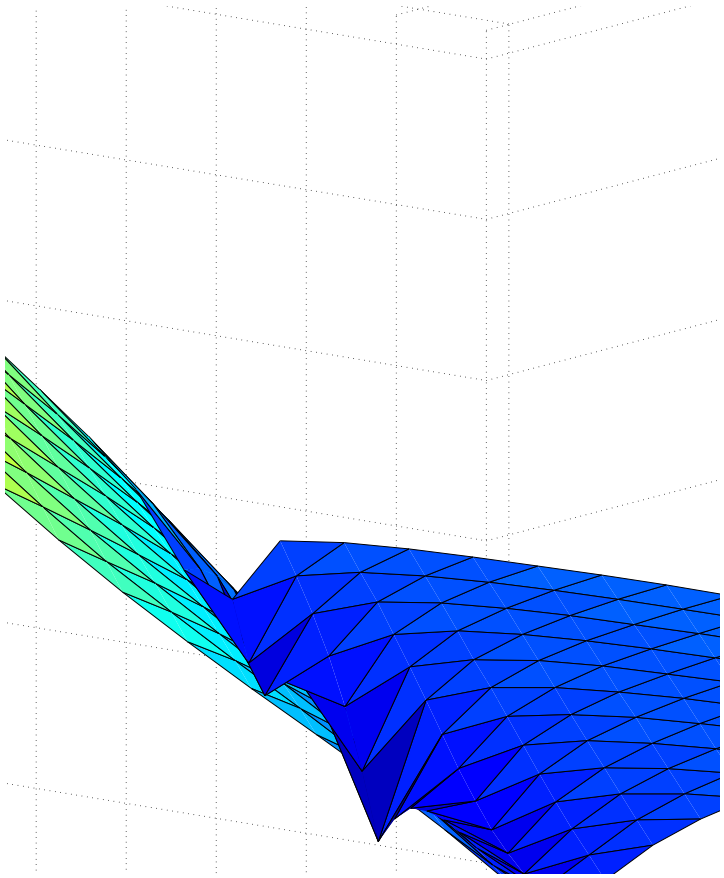


Figure 38: *Zoom on one part of the pseudo-minimum "valley" from the surface of F*

If we consider a starting point close to the global minimum, such as (0.8, 0.13) for instance, we notice (figure 41) that our algorithm converges towards this global minimum. If we start from a point which is further from the global minimum (such as (1.0, 0.2)), we notice that we do not reach this global minimum (figure 42). This can be understood if we take a look at figure 38, for we then notice the presence of a "valley" of local minima, making the choice of the starting point crucial for the convergence of the algorithm towards a global minimum.

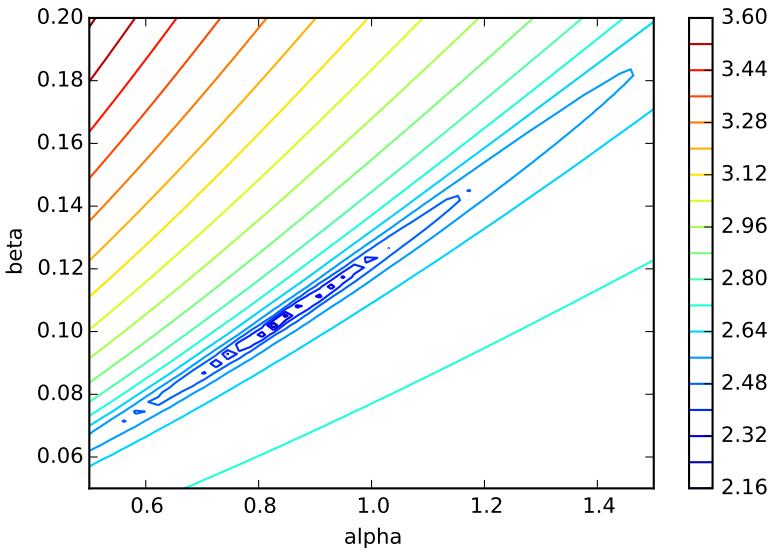


Figure 39: *Contour lines of log(F)*

Besides, we notice that the calculation time is quite important compared to the other algorithms studied. Indeed, for an initial point (0.8, 0.13), the convergence time is about 0.142 s.

Moreover, as it can be seen from figure 43, we notice that for a starting point (1.0, 0.1), the algorithm converges after about 750 iterations (which is a lot, compared to the other optimization algorithms). We also notice a bigger fall as the function gets closer to its extremum.

All these conclusions can be confirmed if we watch figure 44, which shows the trajectory of the parameters α and β on the contour lines of F. Indeed, we see that there is a high number of iterations, and that the algorithm does not necessarily converge towards the global minimum. Besides, we can note the "oscillations" in the value of the parameters, which may explain the lack of efficiency of the algorithm.

More details concerning the comparison of the optimization methods are given in part IV.

Gradient descent with variable step

In order to improve the efficiency of the algorithm and to reduce the

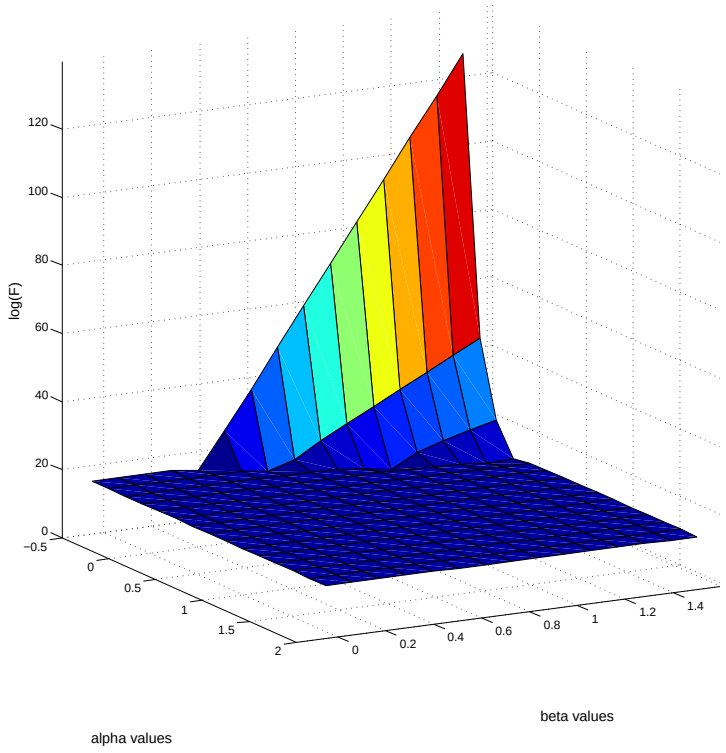


Figure 40: *Extreme values of F for a range of α and β*

calculation time, we start to consider a variable step, which will be iterated with the sequence of points x_i at phase 3 of each iteration. To determine the value of the step, we perform line search, with the Fletcher-Lemarechal algorithm, and the Wolfe conditions.

Theory

The main issue with the gradient descent with constant step is that the step does not adapt with the context and the course of the function. This may be a problem: indeed, on the one hand, a too large step value will have a hard time getting closer, with accuracy, to the global

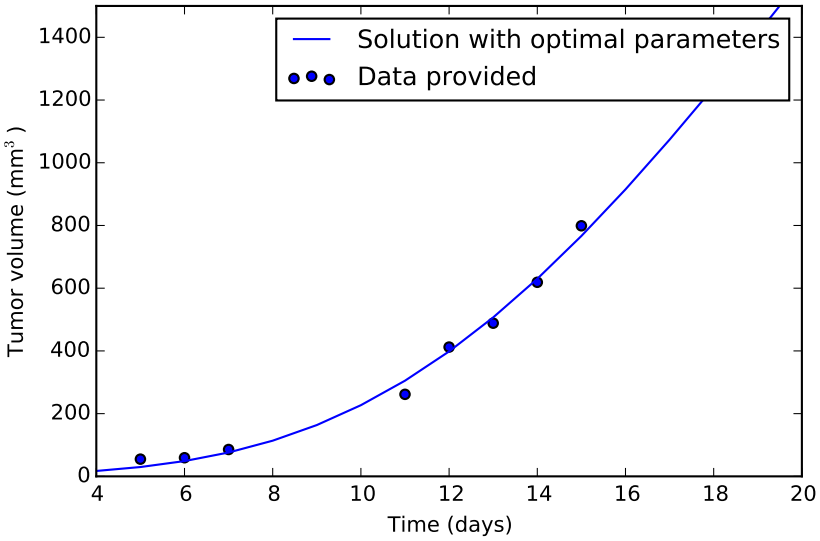


Figure 41: *Solution obtained for the optimized parameters, starting from* $(0.8, 0.13)$ *(gradient with constant step).*

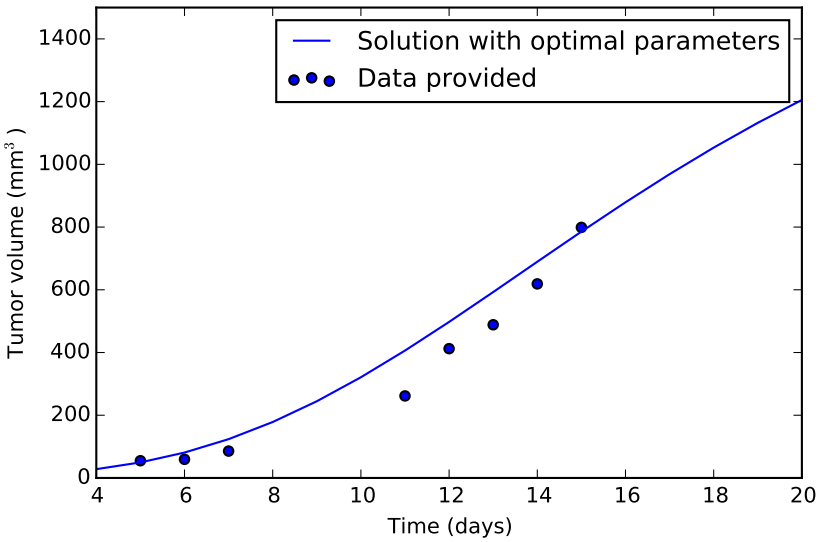


Figure 42: *Solution obtained for the optimized parameters, starting from* $(1.0, 0.20)$ *(gradient with constant step).*

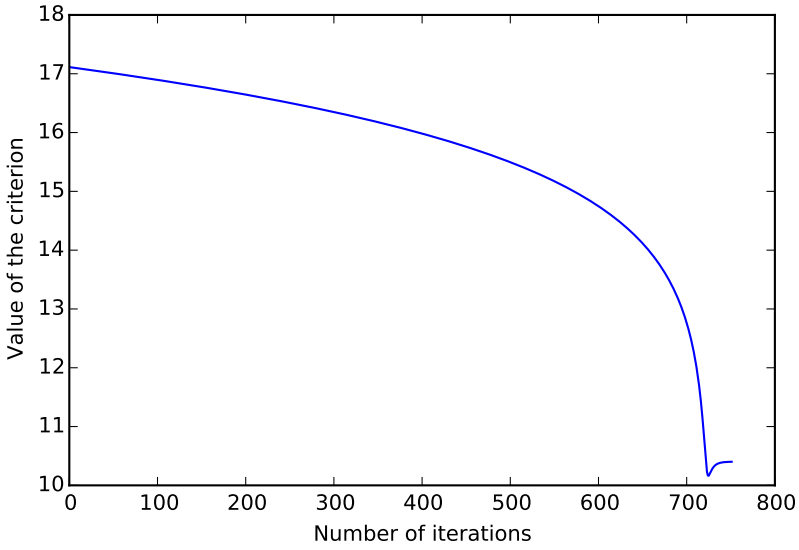


Figure 43: *Value of the criterion as the constant gradient algorithm is carried out (depending on the number of the considered iteration)*

minimum, since every iteration will keep the point to a non-negligible distance from the minimum. But on the other hand, a too small step might converge very slowly (or not converge towards an extremum !) and lead to a tremendous calculation time. The problem being that "too small" or "too large" is not a global point of view, but a local one ; at each iteration, the algorithm must make sure that the step, compared to the context of the function, remains moderate.

These two check-up in our algorithm are carried out at each iteration through the 2 Wolfe conditions, defined as follows (where α_k is the considered step, d_k is the descent direction, which is for the gradient algorithm $-\nabla F(x_k)$, ω_1 and ω_2 real values such as $0 < \omega_1 < \omega_2 < 1$):

$$F(x_k + \alpha_k d_k) - F(x_k) \leq \omega_1 \alpha_k \langle \nabla F(x_k), d_k \rangle \tag{22}$$

$$\langle \nabla F(x_k + \alpha_k d_k), d_k \rangle \geq \omega_2 \langle \nabla F(x_k), d_k \rangle \tag{23}$$

The first Wolfe condition (4), called "sufficient decrease condition" or "Armijo rule", forces the step α_k to reduce sufficiently the optimization criterion F between two iterations of the optimization algorithm.

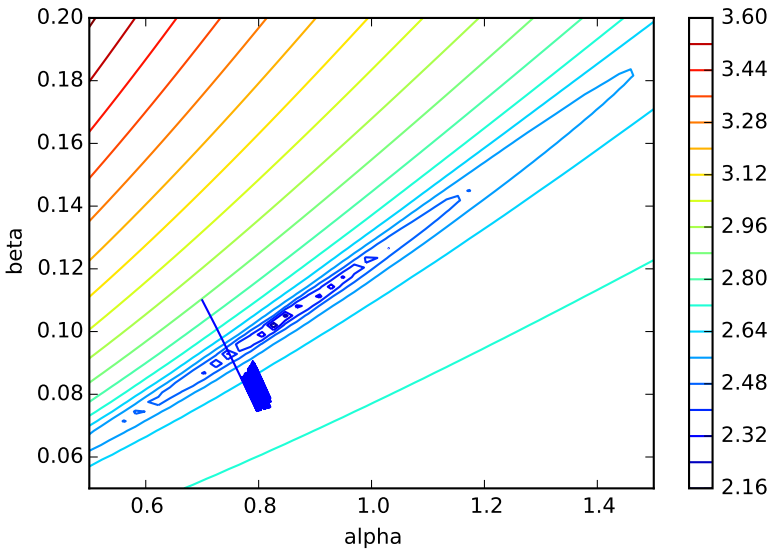


Figure 44: *Value of the parameters α and β in the course of the constant gradient algorithm, compared with the contour lines of* $\log(F)$ *for a starting point (0.7, 0.1)*

It can be noted in this rule that for a minimization problem, both the left-hand term and right-hand terms are negative (for d_k is a descent direction). Hence, this condition states that the choice of the step value α_k must induce a decrease in the optimization criterion at least equal to $\omega_1 \alpha_k \langle \nabla F(x_k), d_k \rangle$. This "sufficient decrease condition" forces to choose a step value that ensures that a minimal decrease in F actually takes place.

The right-hand term (the minimal decrease in F) depends on the gradient of F, which means that the minimal decrease is decided locally: if the gradient is small, the condition will expect a small minimal decrease in F, and if it is large, then we can expect a larger decrease in F. More precisely, we can understand the "sufficient decrease condition" as: the step α_k must be chosen so that the minimal decrease in F between two iterations is at least a fraction ω_1 of what would happen if we could neglect all second-order terms in the Taylor development of F. This specific choice of right-hand term is interesting, for, as said before, it involves the presence of the gradient of F, but also because it guarantees that for some values of step, the condition is necessarily satisfied. Indeed, when α_k is "small enough", then the second order

terms in the Taylor development can be neglected. Therefore, we have the equality:

$$F(x_k + \alpha_k d_k) - F(x_k) = \alpha_k \langle \nabla F(x_k), d_k \rangle + o(||\alpha_k^2||). \qquad (24)$$

Since the right-hand term is negative, if we multiply it by $\omega_1 < 1$ we indeed get a value which is larger than the difference in F.

The point of this rule is that it chooses a step α_k that allows a minimal decrease in F between 2 iterations. This minimal decrease is defined locally (gradient at this point), and there is always a step value that satisfies the inequality. There is still a problem: we are sure that a step value satisfies the equation, but as we have seen before, an extremely small step can then be chosen. This might of course reduce the convergence speed. For this reason, this rule is combined with the second Wolfe condition, called the "curvature condition" (5).

This second condition ensures that the step α_k chosen is "large enough". The idea is this time to check the directional derivative of F (depending on the descent direction d_k) for the two successive values of considered points. If the difference is not large enough, which implies that the step is too small (the two successive points are too close), inequality is not satisfied. This allows us to solve the problem caused by the previous rule, of a too small step value.

To calculate the proper step, and use the two previous Wolfe conditions, we use the Fletcher-Lemarechal algorithm, which is described below:

(a) We first consider two values α_b and α_h, which will represent respectively the lower boundary, and the upper boundary of our possible step value. They will be updated as the algorithm is carried out. We start with $\alpha_b = 0$, $\alpha_h = +\infty$. We consider an initial step α.

(b) We check if, with the step α, the first Wolfe condition is satisfied. While this is not the case, we proceed to the following instructions: $\alpha_h = \alpha$, $\alpha = \frac{\alpha_b + \alpha_h}{2}$.

(c) The first condition is satisfied, now we check whether the second condition is satisfied. If not, we choose $\alpha_b = \alpha$. We have two possibilities: if $\alpha_h = +\infty$ (the first condition was directly satisfied), then $\alpha = 2\alpha_b$. Else, we pick $\alpha = \frac{\alpha_b + \alpha_h}{2}$. We check if the

second condition is satisfied with this step ; if it is not the case, we restart this phase.

(d) We return the step α obtained: it is an optimal step.

Analysis of the results

This algorithm was implemented, and applied to the function $\log(F)$ ($\varepsilon = 0.0001$, $\omega_1 = 0.001$, $\omega_2 = 0.99$) for several starting points (figure 45).

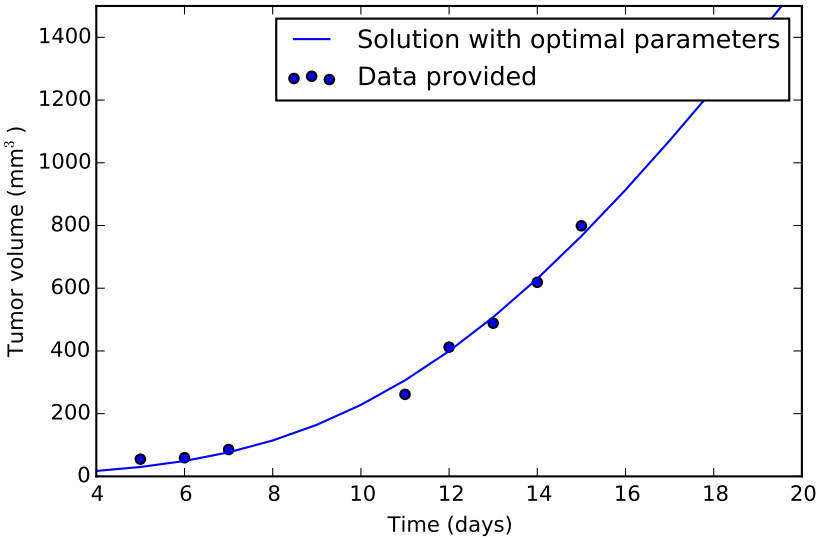


Figure 45: *Solution obtained for the optimized parameters, starting from* $(0.8, 0.13)$ *(gradient with variable step).*

We notice that we are faced with the same problem as in the case of the gradient descent algorithm with constant step: if we choose a starting point too far from the global minimum, then the algorithm will stop at an extremum which is not the global minimum (as illustrated on figure 46).

Though, we notice on figure 47 that the algorithm converges in only 6 iterations for the starting point (0.7, 0.1), and even if each iteration is far more costly than in the case of the constant gradient algorithm, it still leads to a calculation time significantly reduced (divided by 10). Besides, the value of the criterion is slightly diminished as we use this variable gradient algorithm instead of the constant gradient one for

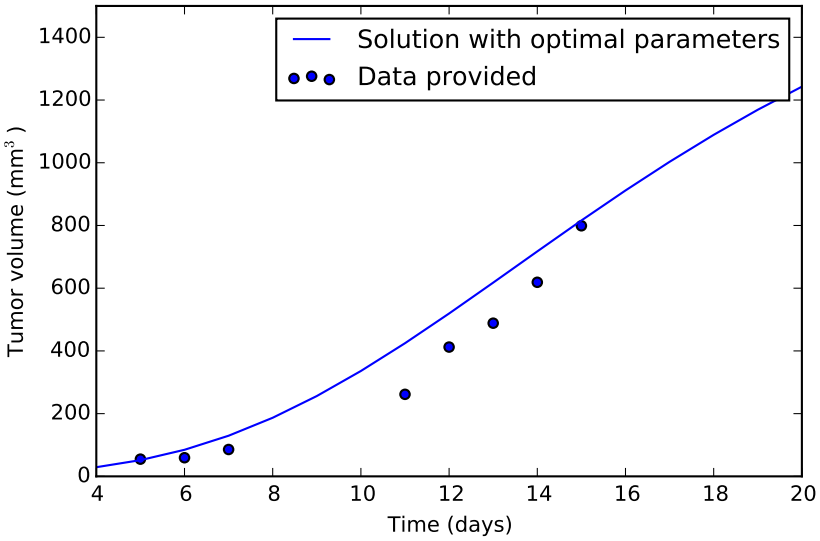


Figure 46: *Solution obtained for the optimized parameters, starting from* $(1.0, 0.20)$ *(gradient with variable step).*

the considered points. This algorithm is therefore to be favored compared with the constant gradient one.

Once more, these results are confirmed if we look at figure 48, which shows the trajectory of (α, β) on the contour lines of $\log(F)$ for the starting point (0.7, 0.1). We can see that the algorithm converges after a very small number of iterations, to a point which is not the global minimum.

More details concerning the comparison of the optimization methods are given in part IV.

The Gauss-Newton algorithm

Even though the variable gradient algorithm seems better than the constant gradient algorithm, an important issue remains: the algorithm does not necessarily return the global minimum that we are looking for. To try and correct this situation, a new algorithm is studied in the following section: the Gauss-Newton algorithm.

Theory

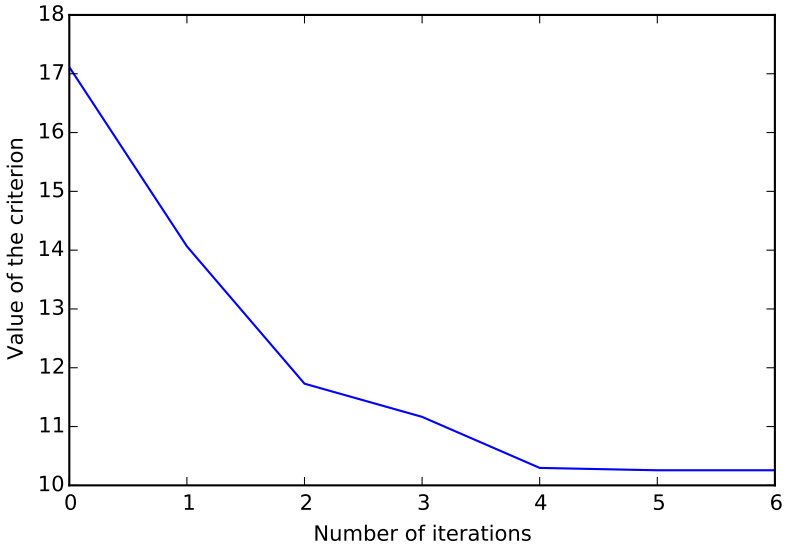


Figure 47: *Value of the criterion as the variable gradient algorithm is carried out (depending on the number of the considered iteration) for a starting point (0.7, 0.1)*

The Gauss-Newton algorithm is an optimization algorithm which is applied on a specific set of problems: the minimization of the sum of squared functions. It is, therefore, totally adapted to our situation.

Let us consider m functions r_i $(i = 1...m)$ (written in a matrix r), depending on n variables $\gamma = (\gamma_1, ...\gamma_n)$. The Gauss-Newton algorithm allows to find the γ that will minimize $\sum_{i=1}^{m} r_i^2(\gamma)$. Here of course, $r_i = y_i - M(t_i; \alpha, \beta)$, $\gamma = (\alpha, \beta)$ and the function that we want to minimize is F. We will call Y the vector $(y_1, ..., y_m)$ afterwards.

The Gauss-Newton method is based on expanding $M(t_i; \gamma)$ as a Taylor series for a small variation in γ, and keeping only the first-order terms. It gives us:

$$M(t_i; \gamma + \delta) = M(t_i; \gamma) + \sum_{j=1}^{n} \frac{\partial M(t_i; \gamma)}{\partial \gamma_j} \delta_j + o(||\delta||^2) \qquad (25)$$

Or, if we write is as a matrix equation $((M_0)_i = M(t_i; \gamma)$, and J_i being

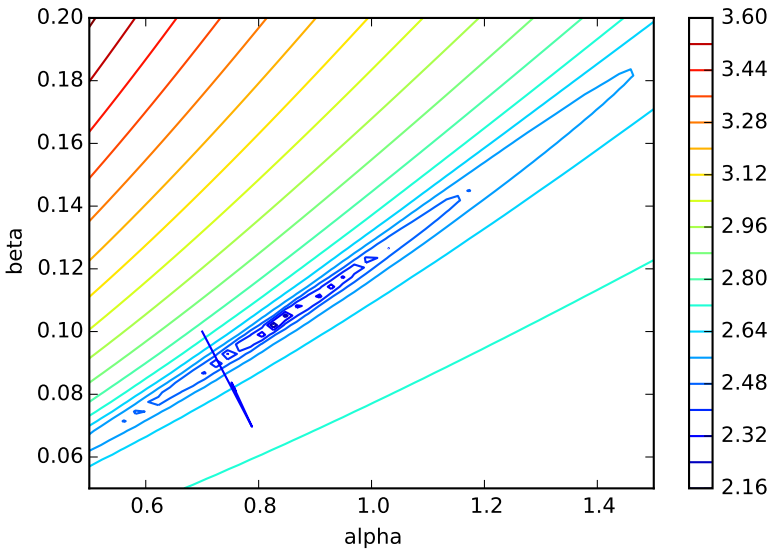


Figure 48: *Value of the parameters α and β in the course of the variable gradient algorithm, compared with the contour lines of* $\log(F)$ *for a starting point (0.7, 0.1)*

the i-th line of J the Jacobian matrix of $M(t;\gamma)$ depending on γ):

$$M(t_i;\gamma+\delta) = (M_0)_i + J_i.\delta \tag{26}$$

We want to select the δ that will minimize F. For this reason, we calculate the δ that will solve the equation $\frac{\partial F}{\partial \delta} = 0$. We obtain:

$$\frac{\partial F}{\partial \delta} = 2\sum_{i=1}^{m} \frac{\partial M(t_i;\gamma+\delta)}{\partial \delta}(y_i - M(t_i;\gamma+\delta)) \tag{27}$$

Or:

$$2\sum_{i=1}^{m} J_i^T(y_i - (M_0)_i - J_i\delta) = 0 \tag{28}$$

It leads to:

$$\sum_{i=1}^{m} J_i^T J_i.\delta = \sum_{i=1}^{m} J_i^T(y_i - (M_0)_i) \tag{29}$$

And if we denote $A = J^T J$, $g = J^T(Y - M_0)$ we find that:

$$A\delta = g \tag{30}$$

When we solve this equation, we find:

$$\delta = (J^T J)^{-1} J^T (Y - M_0) = J^{-1} (J^T)^{-1} J^T (Y - M_0) = J^{-1} (Y - M_0) \tag{31}$$

The previous equation shows us that we find an optimal step which is a generalization of the step employed for the Newton method for a function g depending on only one variable ($\frac{g}{\dot{g}}$).

The algorithm is then defined as follows for an initial point $x = (\alpha, \beta)$ and the required accuracy ε specified in the input:

(a) We check whether $||\nabla F|| > \varepsilon$. If it is not the case, then we return x. Else, we continue onto the next step.

(b) We calculate the coefficients of the vector r applied in x ($r_i = y_i - M(t_i; \alpha, \beta)$) as well as the value of $M(t_i; \alpha, \beta)$ applied at the same point x.

(c) We calculate the Jacobian matrix J of the functions $M(t_i; \alpha, \beta)$.

(d) We calculate the value $\delta = (J^T J)^{-1} J^T r$.

(e) We iterate: $x = x + \delta$.

(f) We restart at phase 1.

We may notice that the algorithm involves the inversion of the Jacobian matrix of the functions $M(t_i; \alpha, \beta)$. If at one iteration, the matrix is not invertible, or ill-conditioned, the algorithm can encounter difficulties.

Analysis of the results

This algorithm was implemented, and applied to the function F (with an accuracy equal to 0.01) for several starting points.

We noticed this time that the algorithm converges to the global minimum, for a range of initial values far wider. Indeed, starting from $(1.7, 0.2)$, or $(1.0, 0.13)$, we ended up each time at the global minimum of the function (figures 49 and 50). Besides, the calculation time was, in the two cases, very little compared to the calculation times for the

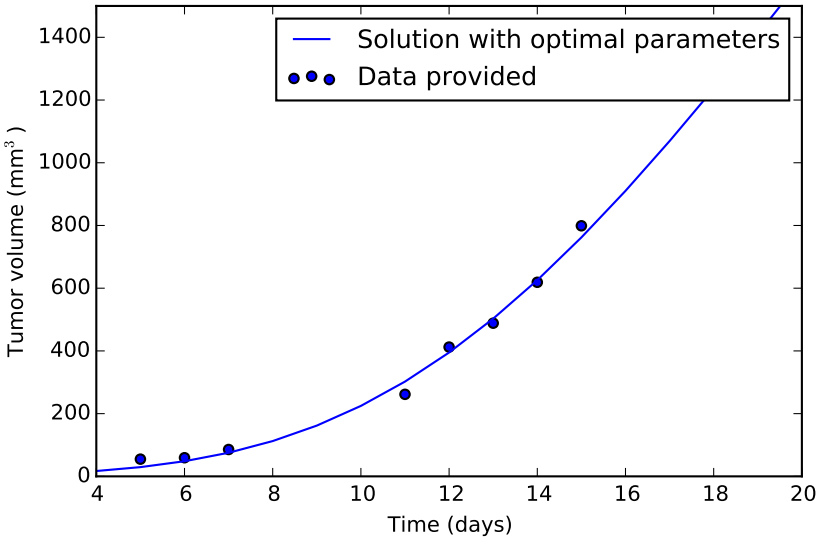


Figure 49: *Solution obtained for the optimized parameters, starting from* $(1.7, 0.2)$ *(Gauss-Newton).*

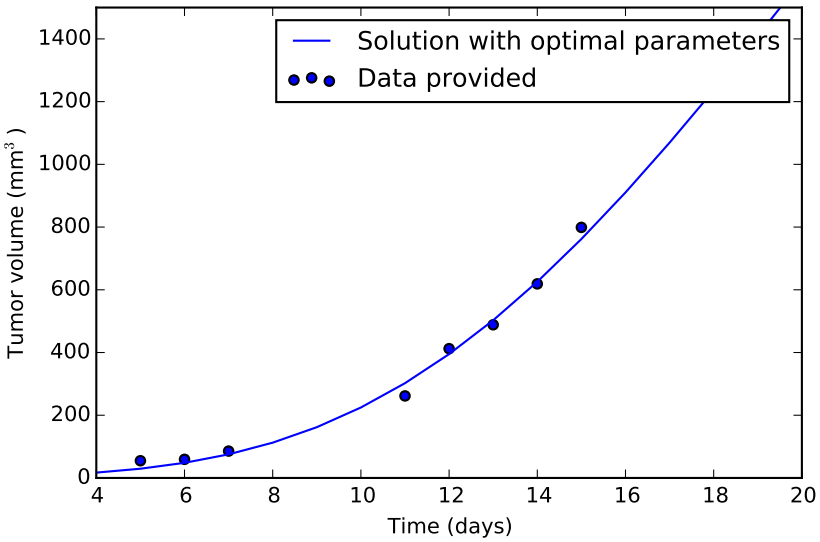


Figure 50: *Solution obtained for the optimized parameters, starting from* $(1.0, 0.13)$ *(Gauss).*

two other algorithms. Indeed, the Gauss-Newton algorithm leads in this case to a calculation time inferior to 0.0001 s, which is really low when compared with the previous algorithms considered.

Plotting the value of the criterion when the Gauss-Newton algorithm is applied depending on the number of the considered iteration, we notice several elements (figure 51). First, that the algorithm converges very quickly (only 8 iterations), and that it indeed converges to a far better criterion value than the two previous algorithms.

These results are confirmed in figure 52, where we can see that the algorithm converges towards the global minimum in a small number of iterations.

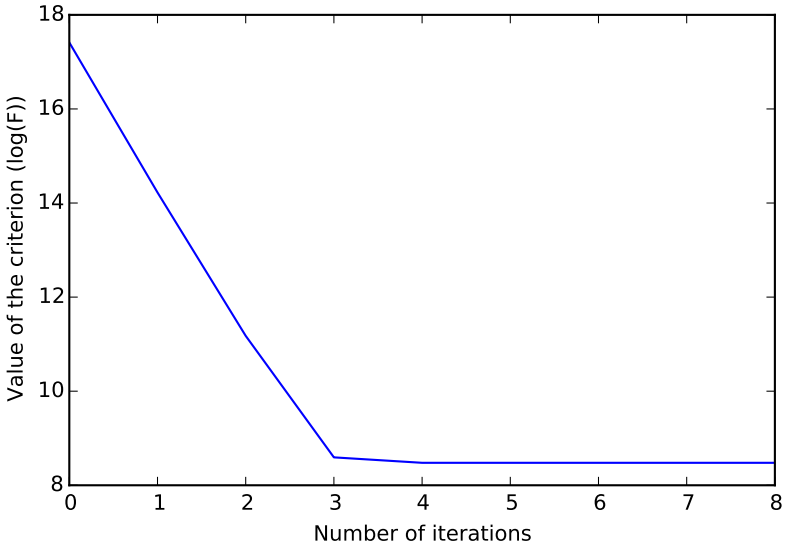


Figure 51: *Value of the criterion as the Gauss-Newton algorithm is carried out (depending on the number of the considered iteration)*

But, with a starting point $(0.8, 0.13)$, we notice (figure 53) that the algorithm does not converge towards a global minimum at all. Indeed, the final values obtained for α and β are about (-11, -4), very far from the global minimum, and even from the starting point. It is also good to note that with such a starting point, the gradient algorithms reached the global minimum without a difficulty. This shows that the problem of the Gauss-Newton algorithm is a stability one, for it involves a matrix inversion. And if this matrix is ill-conditioned, this

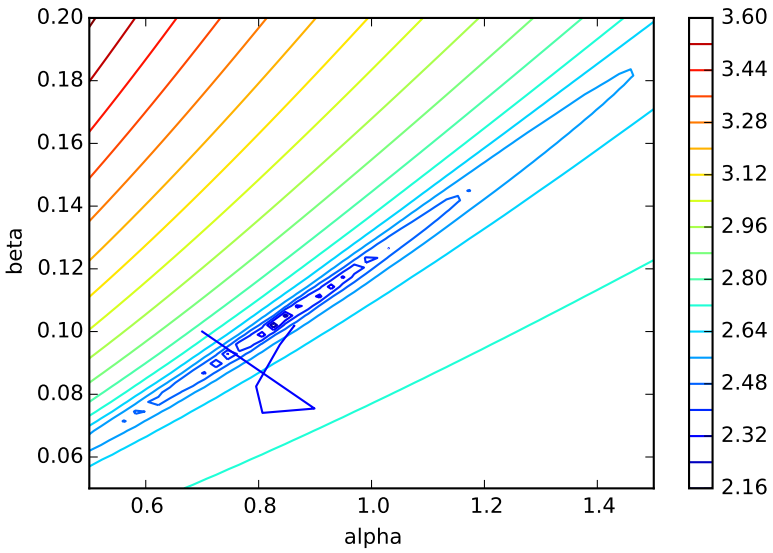


Figure 52: *Value of the parameters α and β in the course of the Gauss-Newton algorithm, compared with the contour lines of* $\log(F)$ *for a starting point (0.7, 0.1)*

may lead to a far higher step value that what would be the optimal step. And since the step value is very high, the approximation carried out in the beginning of the theoretical framework of the algorithm (that all second-order terms in the Taylor development could be neglected) becomes of course a wrong assumption. The algorithm can thus diverge.

The Gauss-Newton algorithm is therefore a very efficient algorithm, but which lacks of stability : some starting points can make the algorithm diverge.

More details concerning the comparison of the optimization methods are given in part IV.

The Levenberg-Marquardt algorithm

The Gauss-Newton algorithm is a very powerful algorithm, but which lacks of stability depending on the starting points. Not all starting points lead to convergence. For that reason, Levenberg and Marquardt ([3], [4]) developed an optimization algorithm which allows to combine

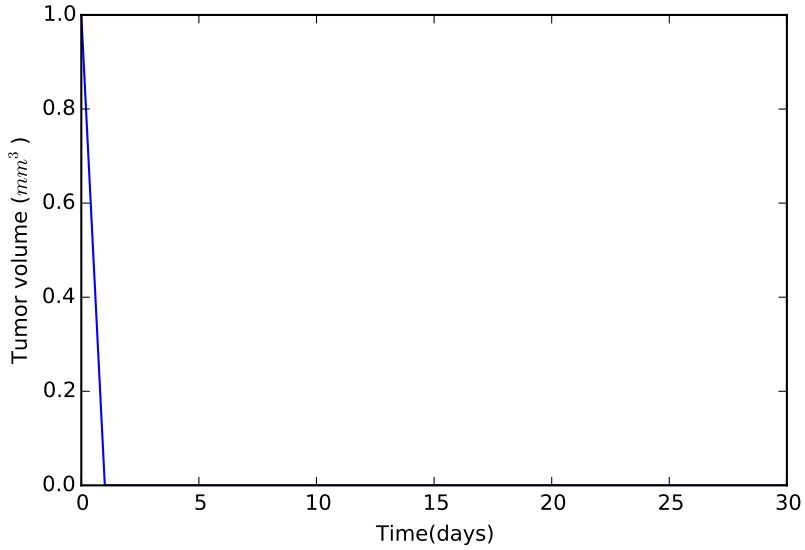


Figure 53: *Solution obtained for the optimized parameters, starting from* $(0.8, 0.13)$ *(Gauss-Newton).*

the efficiency of the Gauss-Newton algorithm and the stability of the gradient algorithms.

Theory

Outline of the method

Let us choose the same notations as the ones used in the section devoted to the Gauss-Newton algorithm. The descent direction δ_t satisfies the following equation in the case of a Gauss-Newton algorithm (as a reminder, $A = J^T J$, $g = J^T(Y - M_0)$):

$$A\delta_t = g \tag{32}$$

We also notice that $g = -\frac{1}{2}\nabla F$. Since the orientation of the direction is the element that will really matter in our algorithm, we will consider from now on that $g = -\nabla F$. Therefore, we can write that, in the case of a gradient descent algorithm, the direction δ_g satisfies the following equation:

$$\delta_g = g \tag{33}$$

The idea of Marquardt in [4] is to try and combine the two previous direction equations, to create a third equation that will be used to calculate the direction δ_0 in this algorithm, with $\lambda \geq 0$:

$$(A + \lambda I)\delta_0 = g \tag{34}$$

The relevance of the parameter λ can be seen quite easily: when λ is close to 0, then the equation that needs to satisfy the direction is very close to equation (14). Therefore, the calculated direction δ_0 will be close to the one calculated for a Gauss-Newton algorithm δ_t. On the opposite, when λ is extremely high, we see that the equation will be closer to equation (15), and that the direction δ_0 calculated will be very close to the opposite of the gradient. This intuitive thought is rigorously shown in [4], where it is even stated that as λ grows, the calculated direction δ_0, starting at δ_t turns more and more towards δ_g. In the same article, is also shown that as λ grows, the norm of the direction decreases.

Relevance of the method

One interrogation that may legitimately arise is whether the direction calculated thanks to equation (16) is a descent direction $\forall\lambda$. Basing ourselves on [4], we will demonstrate properly this through the following theorem:

Theorem : For any positive λ, if δ_0 satisfies the equation (16), then δ_0 minimizes $F(\gamma + \delta)$ on the sphere of radius $||\delta_0||$.

Proof : The function C that we want to minimize can be written as $C(\gamma, \delta) = ||Y - M_0 - J\delta||^2$ for any direction δ. The minimization problem described in the theorem can therefore be written this way:

$$\begin{aligned} \min_{\delta} \quad & ||Y - M_0 - J\delta||^2 \\ \text{subject to} \quad & ||\delta||^2 = ||\delta_0||^2 \end{aligned}$$

To solve the optimization problem, we consider the Lagrangian, with λ a Lagrange multiplier:

$$u(\delta, \lambda) = ||Y - M_0 - J\delta||^2 + \lambda(||\delta||^2 - ||\delta_0||^2) \tag{35}$$

By the method of Lagrange, we know that a stationary point necessarily satisfies the equations $\frac{\partial u}{\partial \delta} = 0$, $\frac{\partial u}{\partial \lambda} = 0$.

Thus, we have:

$$-2J^T(Y - M_0 - J\delta) + 2\lambda\delta = 0 \tag{36}$$

and

$$||\delta||^2 = ||\delta_0||^2 \tag{37}$$

From equation (18), we find:

$$(J^T J + \lambda I)\delta_0 = J^T(Y - M_0) \tag{38}$$

This means that if δ_0 satisfies equation (16), then it is a stationary point for $F(\gamma + \delta)$. Since $A + \lambda I$ is a definite positive matrix, then the calculation of $\frac{\partial^2 u}{\partial \delta^2}$ and of $\frac{\partial^2 u}{\partial \lambda^2}$ shows us that the stationary point δ_0 is indeed a minimum.

This way, we are able to show that the δ_0 direction is indeed a direction that minimizes $F(\gamma + \delta)$.

Rescaling

The solution of Gauss-Newton, δ_t, is invariant under linear transformation, but it is not the case for the solution of the gradient algorithm. For this reason, we will scale the definition space, by considering the standard deviation of the gradient function as unit. More explanations

about this procedure are given in [4]. This leads us to consider the following formulas, where * represents the rescaled values ($a_{j,j'}$ being the coefficients of the matrix A, and g_j the coefficients of the vector g as defined in equation (16)):

$$A^* = (a^*_{j,j'}) = \frac{a_{j,j'}}{\sqrt{a_{j,j}}\sqrt{a_{j',j'}}} \tag{39}$$

$$g^* = (g^*_j) = \frac{g_j}{\sqrt{a_j}} \tag{40}$$

$$\delta_0 = \frac{\delta^*_0}{\sqrt{a_{j,j}}} \tag{41}$$

Evaluation of F

Before describing precisely the method, it is crucial to be able to understand exactly how the λ parameter intervenes. For this, let us suppose we are at the p-th iteration. We then have to solve the equation, and calculate the δ^*_0 such as:

$$(A^* + \lambda^{(p)} I)\delta^*_0 = g^* \tag{42}$$

Once δ^*_0 is calculated, we calculate δ_0 using equation (23). We then consider a new point γ_{new} (in the next part, γ will be called the "starting point"):

$$\gamma_{new} = \gamma + \delta_0 \tag{43}$$

The next step is then to evaluate $F(\gamma_{new})$. The previous algorithm will be what we will refer to when, in the precise description of the algorithm that will be displayed in the next section, we will talk about evaluating F.

Course of the algorithm

The algorithm takes as inputs a real number $\nu > 1$ and the first value of λ considered, λ_0, the accuracy ε and a starting point γ_0. We suppose that we are currently at the p-th iteration of the algorithm (so, we have as values λ_{p-1}, γ_p and F^p, which is the value of F finally obtained for $\lambda = \lambda_{p-1}$ in equation (24) and a starting point γ_{p-1}):

(a) We check whether $||\nabla F|| < \varepsilon$. If it is the case, then we return γ_p. Else, we proceed onto the following phases.

(b) We rescale A and g, as stated in the subsection "Rescaling".

(c) Let us call F_λ^p the value of F finally obtained for $\lambda = \lambda_{p-1}$ in equation (24) and a starting point γ_p. Let us call F_ν^p the value of F finally obtained for $\lambda = \frac{\lambda_{p-1}}{\nu}$ in equation (24) and a starting point γ_p.

(d) We compare F^p and F_ν^p: if $F_\nu^p \leq F^p$, then we set $\lambda_p = \frac{\lambda_{p-1}}{\nu}$, and $F^{p+1} = F_\nu^p$. γ_{p+1} is also valued with the final value of γ in the evaluation of F_ν^p. We then return to phase 1 for a new iteration or the end of the algorithm.

(e) If not, we compare F^p and F_λ^p: if $F_\lambda^p \leq F^p$, then we set $\lambda_p = \lambda_{p-1}$ and $F^{p+1} = F_\lambda^p$. γ_{p+1} is also valued with the final value of γ in the evaluation of F_λ^p. We then return to phase 1 for a new iteration or the end of the algorithm.

(f) Else, we will increase the value of λ. For this, let us call F_w^p the value of F finally obtained for $\lambda = \lambda_{p-1}\nu$ in equation (24) and a starting point γ_p. If $F_w^p \leq F^p$ then we set $\lambda_p = \lambda_{p-1}\nu$ and $F^{p+1} = F_w^p$. Else, we consider F_w^p as the F obtained for a $\lambda^{p-1}\nu^2$, and we compare once more. While $F_w^p > F^p$, we increase λ by multiplying it by ν. When this condition is not verified anymore (let us say it is after w iterations), we set $\lambda_p = \lambda_{p-1}\nu^w$, $F^{p+1} = F_w^p$ and γ_{p+1} is valued with the final value of γ in the evaluation of F_w^p. We then return to phase 1 for a new iteration or the end of the algorithm.

The relevance of the algorithm is quite clear. First, we notice that we consider a new iteration only when the optimization criterion F is reduced, which makes sense in the framework of a minimization problem. But we notice that we do not try to compute the maximal decrease in F between two iterations (indeed, as stated in [4], this local minimal decrease would be a poor choice from a global point of view). Instead, we try whenever it is possible, and when we are sure that it will lead to a local decrease in F, to decrease λ. This is quite logical ; indeed, the smaller λ is, the more the algorithm gets closer from the Gauss-Newton algorithm, which is very efficient as shown in previous parts.

But the point of the algorithm is that when it is not possible to decrease λ without increasing the optimization criterion (typically areas where the Gauss-Newton algorithm would not be stable), then other options are considered. We can keep the same values of λ or, when

there is no other choice if we want to minimize F, increase it until reaching a stage where we have a local decrease of F.

We know that, if we are not already at an extremum (where the condition on the norm of the gradient in the algorithm would be satisfied anyway), we will necessarily reach, for some high value of λ, a local decrease in F. Indeed, for this range value of λ, we are closer to the gradient algorithms, which are stable.

Marquardt, in [4], stated that default values of 0.01 for λ and 10 for ν were often a good choice.

Analysis of the results

This algorithm was implemented and applied to the function F ($\varepsilon = 0.001$, $\nu = 10$, and if not explicitly defined, $\lambda_0 = 0.01$). Applied to the starting point (0.7, 0.1), we notice that, like the Gauss-Newton algorithm, it converges towards the global minimum (figure 54). If we choose a starting point even further from the global minimum, such as (1.7, 0.2) for instance, we notice a convergence towards the global minimum as well (figure 55).

But one interesting feature of the algorithm is that, starting from a point which made the Gauss-Newton algorithm diverge (such as (0.8, 0.13) for instance), we really reach the global minimum required. The Levenberg-Macquardt algorithm seems to have corrected the stability of the Gauss-Newton algorithm, while keeping the ability to converge towards a global minimum from a wide range of starting points (figure 56).

One interesting aspect which can be studied is the evolution of λ as the algorithm goes on. We notice that, depending on the starting point, the evolution will be extremely different. When the starting point is a point that is totally fit for a Gauss-Newton algorithm, then λ decreases very quickly: it is indeed in this case better to get closer to the Gauss-Newton algorithm. On the other hand, when we start with a point such as (0.8, 0.13), which leads to a divergence for Gauss-Newton, λ will increase a lot in the early stages of the algorithm, denoting the fact that it is necessary to copy the properties of the gradient descent algorithms to attain a proper minimum value.

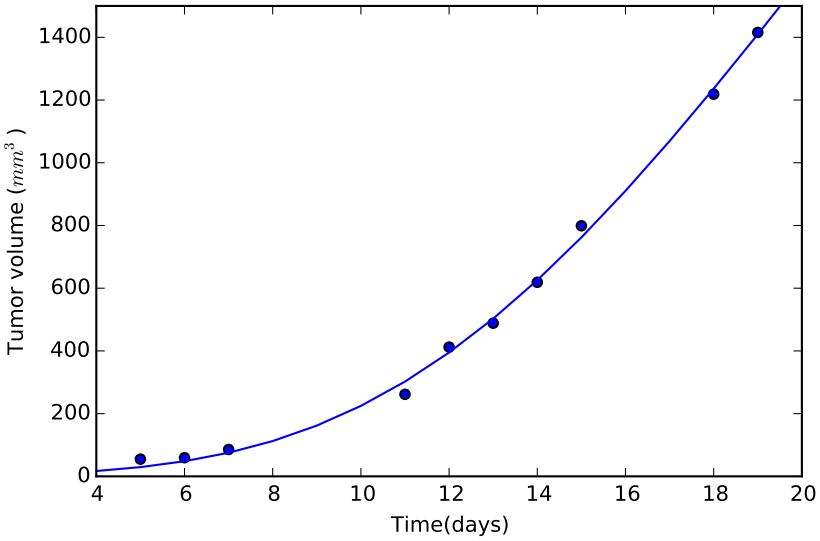


Figure 54: *Solution obtained for the optimized parameters, starting from* $(0.7, 0.1)$ *(Levenberg-Marquardt).*

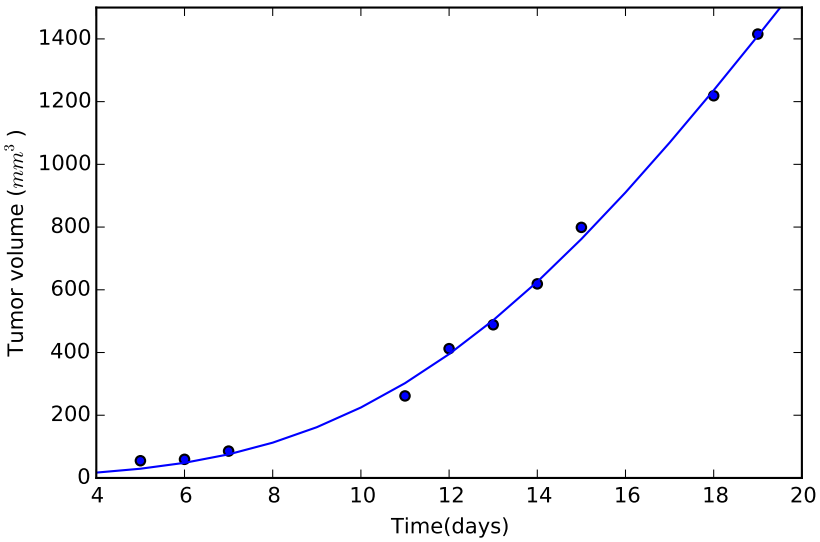


Figure 55: *Solution obtained for the optimized parameters, starting from* $(1.7, 0.2)$ *(Levenberg-Marquardt).*

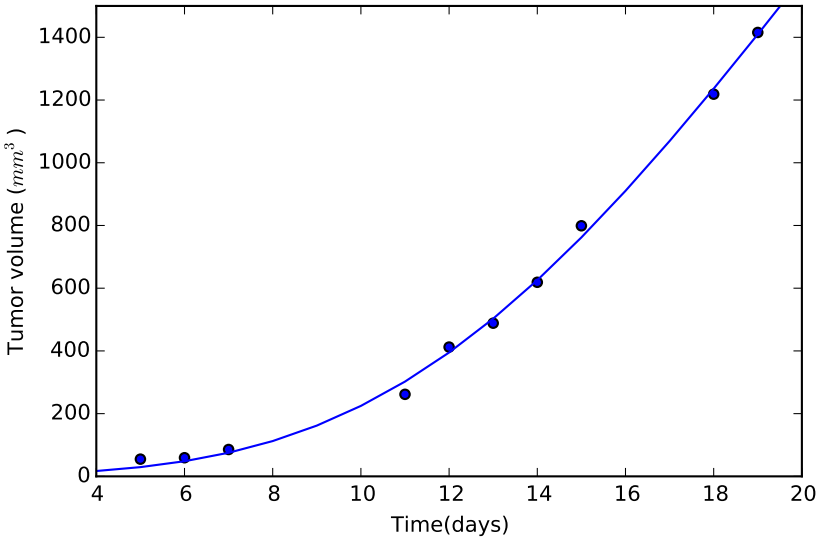


Figure 56: *Solution obtained for the optimized parameters, starting from* $(0.8, 0.13)$ *(Levenberg-Marquardt).*

If we study the calculation time, we notice that it is far better than the gradient descent algorithms for these starting points, but slightly less good than the Gauss-Newton algorithm, which makes perfect sense: in the best-case scenario as far as efficiency is concerned, λ will be close to 0 (but will never really reach it), which will make the algorithm as similar as possible to the Gauss-Newton, without equalizing its outstanding performance in this specific situation.

Yet, the choice of some starting points still leads to numerical problems for the algorithm (notably (1,2)). We notice that the points that seem to challenge the algorithm were also troublesome for the Gauss-Newton algorithm.

Indeed, if we consider figure 39, we notice that the function F can have extreme values, which can create a numerical error if it has to be evaluated by the program. The issue is the evaluation of $F^1\lambda$ and F^1_ν, since the first value λ_0 is given by the user and not, like all the other λ, evaluated by the algorithm depending on the optimization criterion. If the first λ is badly chosen as an input, then we can reach a F which will be impossible to calculate numerically. Indeed, we can see that if we increase the initial value of λ_0 for these troublesome starting points,

then the Levenberg-Marquardt actually converges towards the global minimum. But the fact that the convergence depends on the value of λ_0 is an element that should not be neglected.

If we study the evolution of the optimization criterion (figure 57), and of the parameters related to the contour lines of F (figure 58), we notice a very fast and accurate convergence, even though the algorithm is rather less efficient (but considerably more stable) than the Gauss-Newton algorithm.

More details concerning the comparison of the optimization methods are given in part IV.

8.3.4 Statistical comparison of the optimization algorithms

Theoretical framework

In this part, we will compare the various optimization algorithms more rigorously: instead of comparing the results of the algorithm on a rather restricted number of points, we will apply a statistical procedure in order to get an approximation of the average error (value of the optimization criterion) and calculation time of the algorithm.

To do that, we will use a Monte Carlo method: we will simulate a large number of times the algorithm that we want to study, for a given required accuracy and for various starting points, and we will consider the global calculation time as the average of the calculation time of every simulation. We will do the same thing for the error.

One issue remains: which starting points do we have to consider, so that the results are really significant ? The first matter is: how many points should we consider ? The second one is related to the fact that choosing only random coordinates for the starting points could be a problem. Indeed, since many algorithms (like the gradient descents) have an important dependence on the starting point, a random draw can then lead to very close points, which will not represent well the true properties of the algorithm. For that reason, we tried and perform a statistical analysis more representative.

To carry out the analysis, we followed the advice from [5]. Basing ourselves on the experiments carried out in the article, we decided to perform 100 successive simulations. Besides, as [5] suggests, to choose our

starting points, we used a multi-start strategy based on latin square sampling. This strategy is developed in the following paragraph.

Let us suppose that we want n independent points. The idea of the latin square sampling is to divide the range of values that can take our point coordinates in n smaller domains of equal size. Then, each point is picked randomly within a different domain. This way, we end up getting n points, which are able to represent the whole coordinates range. Besides, since the points are still picked randomly, every statistical analysis will return a different average value.

Results of the statistical analysis

We performed the previous statistical analysis of the 4 optimization algorithms, for the same 100 starting points defined by latin square sampling, an accuracy of 0.01, and we obtained the results displayed on figure 59 (for the Levenberg-Marquardt algorithm, $\lambda_0 = 1000$). The analysis was performed on Python 2.7, on a computer having an Intel Core processor with a 2 GHz frequency and a 16 Go RAM. The initial conditions were chosen within an arbitrary range: [0.5, 1.5] for α and [0.05, 0.2] for β. The function does not have extreme values on this range, hence the decision to choose it.

We noticed several useful pieces of information. First, we saw how hard it is to perform a statistical analysis on the Gauss-Newton method, since its instability leads very often to a numerical error for at least 1 of the 100 simulations. As for the results themselves, we noticed that most of what we had seen and concluded from a reduced set of points seems to be verified. Indeed, the constant gradient algorithm is clearly the slowest algorithm, whereas the Gauss-Newton algorithm is the fastest. As for the error, the gradient descents are far less efficient than the other two algorithms. We also noted that the Levenberg-Marquardt algorithm is the algorithm which combines the best, among the 4 presented algorithms, stability and efficiency.

Still, some surprises arose: the variable gradient algorithm is not more accurate on average than the constant gradient algorithm. This is due to the fact that for some starting points, the variable gradient algorithm will lead to much higher errors than the constant gradient one. But for a high majority of starting points, we still saw that the variable gradient algorithm is seemingly better than the constant gradient one. Moreover, they had a very significant difference as far as calculation

time was concerned.

Another noticeable thing is the difference in accuracy between the Gauss-Newton algorithm and the Levenberg-Marquardt one. This is due to the instability of the first algorithm. Indeed, for some starting points, we reach values which may induce very important errors. But, there again, for nearly all the starting points which do not induce instability, the accuracy obtained is very similar.

The table 59 can quite clearly show us why the Levenberg-Marquardt algorithm is generally favored in the least-squares minimization for biological data. Accurate, robust and fast, it is clearly the best optimization algorithm to solve our problem.

Comparison to a pre-implemented algorithm: the Nelder-Mead algorithm

The Levenberg-Marquardt algorithm was then integrated into a least-squares minimization code used by the team at inria. So far, the optimization algorithm that was used was the Nelder-Mead algorithm. Integrating the Levenberg-Marquardt algorithm into the code was a useful step (the algorithm is not in the standard library of numpy, so having a functional code of it was helpful), and was also the opportunity to compare the performance of the two algorithms for the same data set. In this section, the considered data set is independent from the data studied before.

A statistical analysis analogous to the one carried out in the previous part (for, when not specified, $\lambda_0 = 100$ and an accuracy of 10^{-6} for Levenberg-Marquardt) led to the following results. On the studied data, the average calculation time of the Levenberg-Marquardt algorithm was slightly more significant. As for the average error committed, we noticed that the Nelder-Mead algorithm was a little less accurate, but the difference is extremely small (and could almost be neglected): we have indeed an average difference of $2.31.10^{-6}$ for default parameters.

As for the parameters of Levenberg-Marquardt, obviously, reducing the required accuracy increases the error, and reduces the calculation time. Increasing the starting value of λ_0 increases the calculation time, but if it is too small, the error sharply increases. All these results are summed up on table 60. If we have a negative value in the second column (average relative difference between the method and the

Nelder-Mead algorithm), then this means that for those parameters, the Levenberg-Marquardt algorithm is less efficient than the Nelder-Mead algorithm. In this figure, LM stands for Levenberg-Marquardt and N-M for Nelder-Mead.

8.3.5 Numerical implementation of the Levenberg-Marquardt algorithm

All these algorithms were coded using Python 2.7. The previous analysis showed that the Levenberg-Marquardt algorithm was the most efficient and interesting to use.

For this reason, an easy-to-handle version of the Levenberg-Marquardt function had to be implemented from the version that had already been coded within the framework of the previous analysis. The written code was integrated into a research numerical code developed by the MONC team at inria, aiming at the analysis of preclinical data in oncology. This part of the work was quite challenging, for it required a clean-up, series of tests and an optimization of the code, so that a user who did not know how it was coded could find it easy to use and efficient, and for a good understanding of the research code was necessary.

To integrate it into the code, several elements had to be added or modified. Among them, the functionality of automatic calculation of the Jacobian matrix through finite differences was added. The various Levenberg parameters were also grouped within a class. The code was finally integrated into the research numerical code, and was functional.

8.4 Numerical methods for the Gompertz model

8.4.1 Introduction

Setting of the problem

As of today, the phenomenon of tumor growth is still an open problem. Indeed, no equation is considered as the equation which would perfectly model the data. Yet, for, notably, biological reasons, the Gompertz model is one of the most acclaimed models to describe this growth ([1], [2]). During this project, a simplified model of tumor growth was considered, deriving from the Gompertz model, from which an exact solution can be found rather easily . The work was to try several numerical methods to see how close their results were to the exact solution, and how various parameters affected this difference. This work is interesting for the results that were found about the efficiency of various numerical methods for a rather simple model can be extended to more complicated ones.

All the following results were computed using Python and its numerical (numpy) and graphical (matplotlib) libraries.

The Gompertz model

The Gompertz model is a model which is commonly used in the literature to describe tumor growth. It can be written as follows:

$$\begin{cases} \dfrac{\partial V}{\partial t} = (\alpha - \beta \ln(V))V \\ V(0) = V_0 \end{cases} \tag{44}$$

We notice that the form of the solution curve (figure 61) is compatible with a phenomenon of tumor growth. During the early stages of cancer, there are very few tumor cells, restricting, therefore, the growth rate. Once the number of tumor cells rises, the growth rate will undoubtedly increase until a certain stage. Indeed, since the resources for all cells are limited, the scarcity of the nutrients when there are too many tumor cells tends to gradually decrease the growth rate. This function seems to fit well the phenomenon of tumor growth.

α and β (and sometimes even V_0) are usually parameters which are not given, and which have to be found through estimations. Since what

we want in this project is to evaluate the accuracy of several methods, we will consider a reference, fixed model with $\alpha = 1$, $V_0 = 1mm^3$ and $\beta = 0.1$. Besides, it was decided to consider a time span of 30 days.

8.4.2 Analysis of numerical methods for the Gompertz differential equation

Runge-Kutta method

Description of the method

The Runge-Kutta method (which will be called RK4 from now on) is an iterative method, which allows to find the approximation of time-depending solutions y of ordinary differential equations (which can be written as follows):

$$\begin{cases} \dot{y} = f(t, y) \\ y(t_0) = y_0 \end{cases} \tag{45}$$

The RK4 method is defined, for a step h, a time span [t_0, t_n] and $\forall i \in \{0, ..., n\}$, an approximation $y_i = y(t_i)$, such as:

$$\begin{cases} y_{i+1} = y_i + \dfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ t_{i+1} = t_i + h \end{cases} \tag{46}$$

where:

$$\begin{cases} k_1 = f(t_i, y_i) \\ k_2 = f(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}k_1) \\ k_3 = f(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}k_2) \\ k_4 = f(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}k_3) \end{cases} \tag{47}$$

The RK4 method is a fourth-order method ; the error is on the order of $\mathcal{O}(h^4)$.

Comparison of the RK4 approximation to the exact solution

We then computed the RK4 method and plotted the approximation obtained, for various steps h (figures 62, 63, 64).

We calculate the relative error L^∞ (the maximal relative difference between the values of the exact solution, and the one of the RK4 approximation). We obtain the results displayed figure 65.

We notice of course that the relative error decreases with the step, but that the calculation time rises, for a smaller step involves more calculations. We also notice that the relative error decreases very quickly with the step. Therefore, even a rather important step value (2) can lead to very accurate results.

Euler method

Description of the method

Using the same notations as the ones that were used in the previous part, the approximations y_i are defined, for the Euler method, as:

$$\begin{cases} y_{i+1} = y_i + hf(t_i, y_i) \\ t_{i+1} = t_i + h \end{cases} \tag{48}$$

The Euler method is a first-order method ; the error is on the order of $\mathcal{O}(h)$.

Comparison of the Euler approximation to the exact solution

We then computed the Euler method and plotted the approximation obtained, for various steps h (figures 66, 67, 68). We also calculated the error (defined as before) figure 69.

We may notice that the Euler method is clearly less accurate than the RK4 method to approximate the solution. It is less accurate, but the calculation time is shorter.

Comparison of the two methods

We now try and compare the two methods. For this, we seek for the maximal step that we can choose in each method to obtain a relative error inferior to various tolerance thresholds. These steps being chosen,

we compare the calculation time, and we can therefore find the faster method to reach a given relative error.

We choose 3 error thresholds (10 %, 1 %, 0.1 %) and we compute a heuristic method which allows us to approximate each time the bigger step that leads to a relative error inferior to these thresholds. Then, by performing several tests in the neighborhood of the step given by the heuristic, we end up finding the bigger step that leads to this specific error threshold. We also write the calculation time. We get the results displayed on figure 71.

We notice that for the same relative error, the RK4 method is faster than the Euler method. Nevertheless, we also see, by comparing figures 65 and 69, that for a given step value (2 for instance), without considering the accuracy of the method at all, the Euler method is considerably faster. This seems to indicate that when the same number of steps is performed in the two methods, the Euler method is faster, which leads to the logical conclusion that one iteration in the Euler method is considerably faster than one iteration in the RK4 method. The RK4 method is nevertheless to be favored ; this is due to the fact that the number of steps that need to be carried out in the RK4 model to attain a certain accuracy is far lower than for the Euler method, since the step value is higher.

To approximate this type of problem, it is therefore better to use the RK4 method.

8.4.3 Derivation of the order of the methods

In this part, we try and determine the order of the two methods employed from the calculations that we have performed.

Theoretical reasoning

The consistency error of a numerical method is a sequence which can be defined as (where y is the exact solution of the ODE):

$$e_n = y(t_{n+1}) - y_{n+1} \tag{49}$$

A method is defined as a p-th order method if $e_n = o(h^{p+1})$.

The simplest way to evaluate the order of a numerical method is to consider the following relationship, deduced from the definition of e_n:

$$\ln(y(t_{n+1}) - y_{n+1}) = \ln(e_n(h)) \tag{50}$$

So, if we plot $\ln(y(t_{n+1}) - y_{n+1}))$ depending on $\ln(h)$, we should obtain a line, the slope being the order of the method.

Numerical results

Order of the RK4 method

We plot this graph, and we obtain the result displayed figure 72.

A linear interpolation gives a slope coefficient of 3.71. We find, indeed, that the RK4 method is a fourth-order method.

Order of the Euler method

The same thing is done for the Euler method, with the result displayed figure 73.

A linear interpolation gives a slope coefficient of 0.87. We therefore find, indeed, that the Euler method is a first-order method.

9 Thanks and acknowledgments

This internship was, for me, an amazing experience, and I would really like to thank Sebastien for his patience throughout the internship (and long after), for his kindness, support, guidance and good advice throughout the six months. Scientifically and humanly, this internship was an extremely positive experience for me, and I am very grateful to Sebastien for it.

I would also like to thank Thierry Colin for accepting me into the team and for his everyday human warmth. I am also very grateful to Olivier Saut for his guidance and help throughout those six months. I would

also like to thank Etienne, Thomas and Cynthia, for their kindness, and their contribution to some of the scientific contents of this report. Finally, I would also like to thank all other members of the monc team for their warmful welcome in Bordeaux.

10 Bibliography

[1] Benzekry S., Lamont C., Beheshti A., Tracz A., Ebos J., et al. (2014) Classical Mathematical Models for Description and Prediction of Experimental Tumor Growth. *PLoS Computational Biology*, 10(8): e1003800. doi:10.1371/journal.pcbi.1003800

[2] Gerlee P. (2013) The model muddle: in search of tumor growth laws. *Cancer Research*, 73(8), 2407-11.

[3] Levenberg K. (1944) A method for the solution of certain non-linear problems in least squares. *Quarterly Appl Math* 2:1648

[4] Marquardt D. (1963) An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, Vol 11, no2, 431-441

[5] Raue A., Schilling M., Bachmann J., Matteson A, Schelke M, et al. (2013) Lessons Learned from Quantitative Dynamical Modeling in Systems Biology. *PLoS*

[6] Ebos J., Lee C., Crus-Munoz W., Bjarnason G., Christensen J. (2009) Accelerated metastasis after short-term treatment with a potent inhibitor of tumor angiogenesis. *Cancer Cell*, 15 :232-239.

[7] Benzekry S. (2011). *Modélisation et analyse mathématique de thérapies anti-cancéreuses pour les cancers métastatiques*, PhD thesis, Université d'Aix-Marseille.

[8] Ebos J., Mastri M., Lee C., Tracz A., Hudson J. et al. (2014), Neoadjuvant antiangiogenic therapy reveals contrasts in primary and metastatic tumor efficacy, *EMBO Molecular Medicine*

[9] Hanahan D., Weinberg R. (2000). The Hallmarks of Cancer, *Cell.*

[10] Iwata K., Kawasaki K., Shigesada N. (2000). A dynamical model for the growth and size distribution of multiple metastatic tumors, *Journal of theoretical biology*, 203, 177-86.

[11] Wheldon T. (1988) *Mathematical models in cancer research.*

[12] Camillo A. (2014), *An experimentally-based modeling study of the effect of anti-angiogenic therapies on primary tumor kinetics for data*

analysis on clinically relevant animal models of metastasis, internship report.

[13] Morais C. (2014) Sunitinib resistance in renal cell carcinoma. *Journal of Kidney Cancer and VHL.*

[14] Simeoni M., Magni P., Cammia C. et al (2004). Predictive pharmacokinetics-pharmacodynamic modeling of tumor growth kinetics in xenograft models after administration of anticancer agents. *Cancer research.*

[15] Dhillon S. (2006) *Clinical pharmacokinetics.*

[16] Aouba A. (2007). Les causes médicales de décès en France et leur évolution 1980-2004. *Bulletin épistémiologique hebdomadaire*

[17] Hahnfeldt P., Panigrahy D., Folkman J., Hlatky L. (1999). Tumor Development under Angiogenic Signaling: A Dynamical Theory of Tumor Growth, Treatment Response, and Postvascular Dormancy. *Cancer Research*, 4770-5.

[18] Benzekry S., Tracz A., Mastri M., Corbelli R., Barbolosi D., Ebos J. (2016). Modeling Spontaneous Metastasis following Surgery: An In Vivo-In Silico approach. *Cancer Research*, 76(3), 535-547.

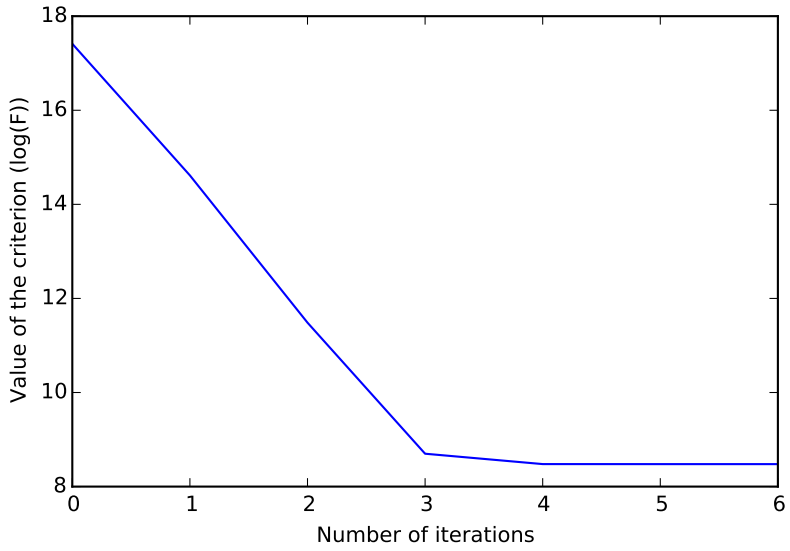


Figure 57: *Value of the criterion as the Levenberg-Marquardt algorithm is carried out (depending on the number of the considered iteration) for a starting point (0.7, 0.1)*

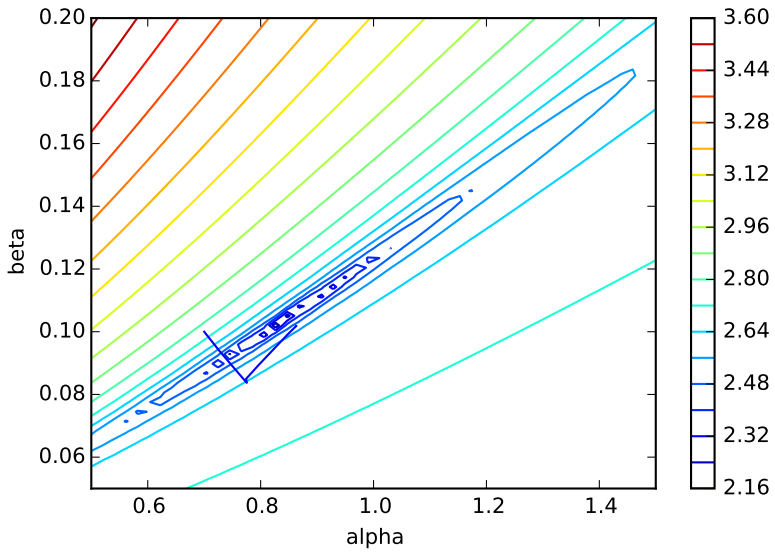


Figure 58: *Value of the parameters α and β in the course of the Levenberg-Marquardt algorithm, compared with the contour lines of* $\log(F)$ *for a starting point (0.7, 0.1)*

Method	Average error ($\log(F)$)	Average calculation time (s)
Constant gradient	14.729	0.2286
Variable gradient	15.629	0.0079
Gauss-Newton	10.423	0.0024
Levenberg-Marquardt	8.478	0.0073

Figure 59: *Average error and calculation time for the considered optimization algorithms*

Method and parameters	Relative difference with the NM method (%)	Calculation time (s)
L-M (default values)	$2.31.10^{-6}$	2.67
L-M (accuracy: 1E-3)	$-2.82.10^{-6}$	2.66
L-M ($\lambda_0 = 1000$)	$6.81.10^{-6}$	2.71
L-M ($\lambda_0 = 0.01$)	-81,56	2.67
N-M	/	2.64

Figure 60: *Results obtained, by comparison with the Nelder-Mead algorithm*

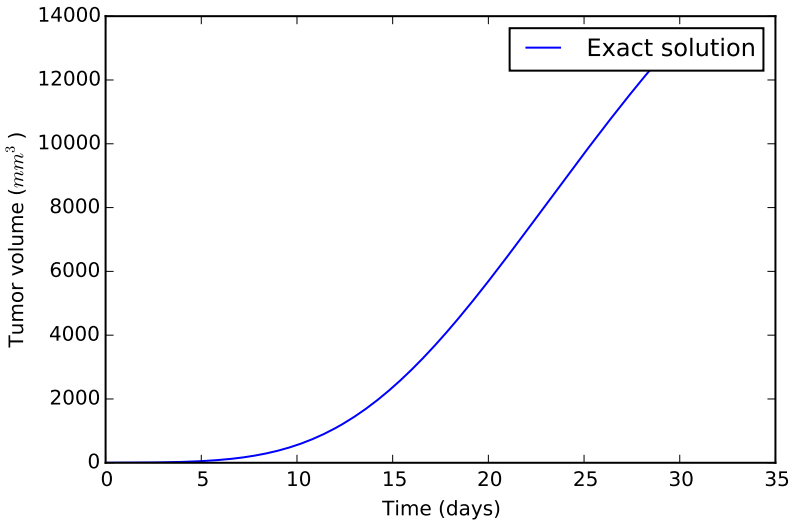


Figure 61: *Solution to the Gompertz differential equation, with* $V_0 = 1mm^3$, $\alpha = 1$, $\beta = 0.1$

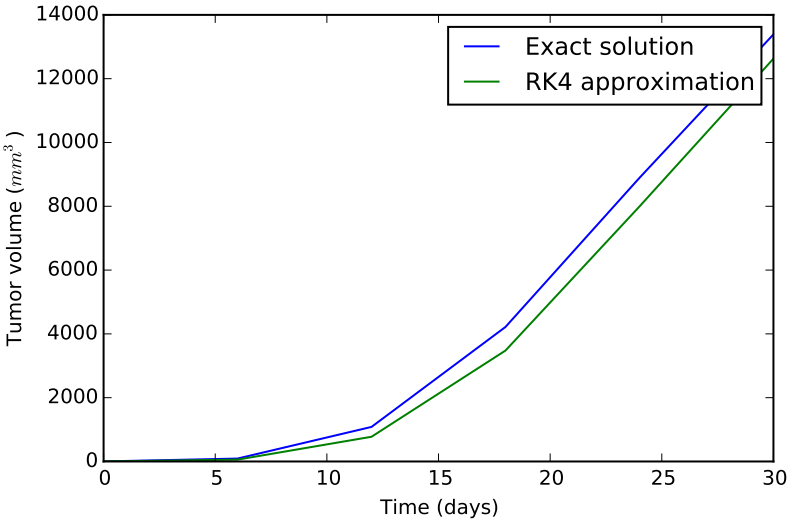


Figure 62: *RK4 approximation for* $h = 6$.

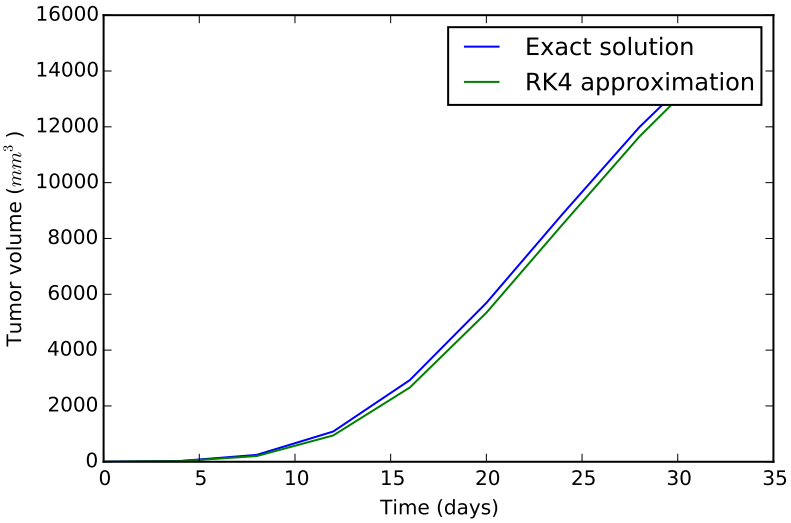
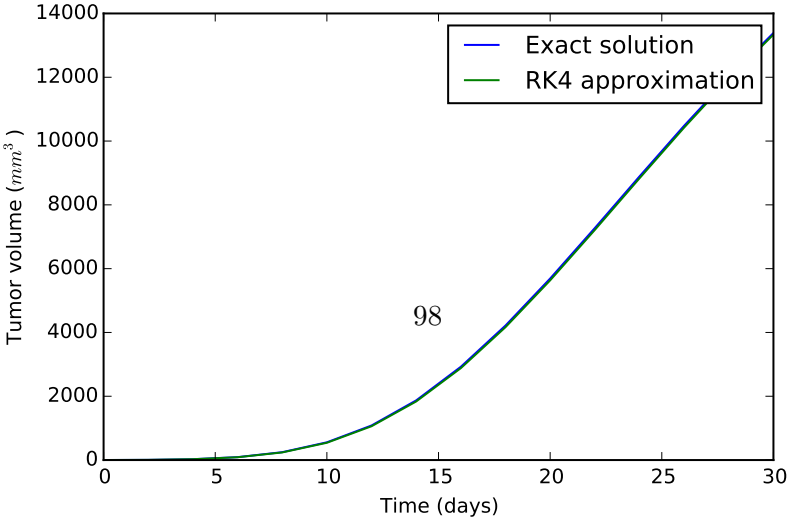


Figure 63: *RK4 approximation for* $h = 4$.



Value of the step	Relative error	Calculation time (s)
6	10.1 %	< 0.0001
4	4.2 %	< 0.0001
2	0.7 %	0.001

Figure 65: *Relative errors and calculation times for the RK4 approximation depending on the value of the step h*

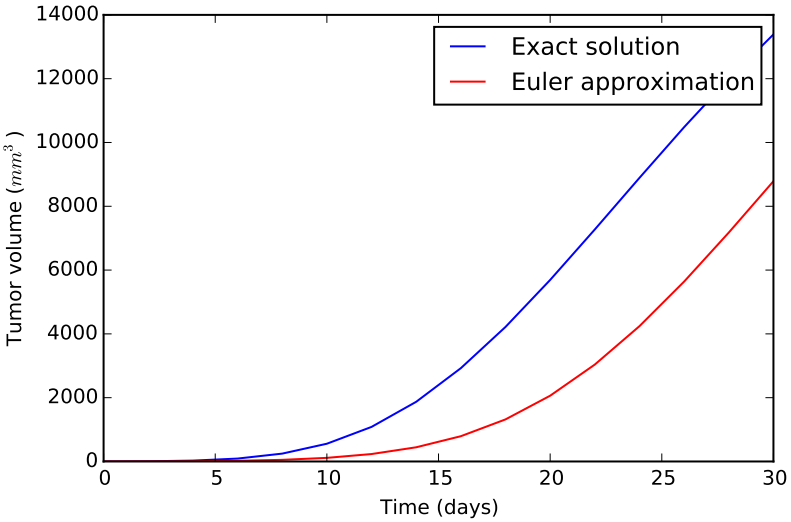


Figure 66: *Euler approximation for* $h = 2$.

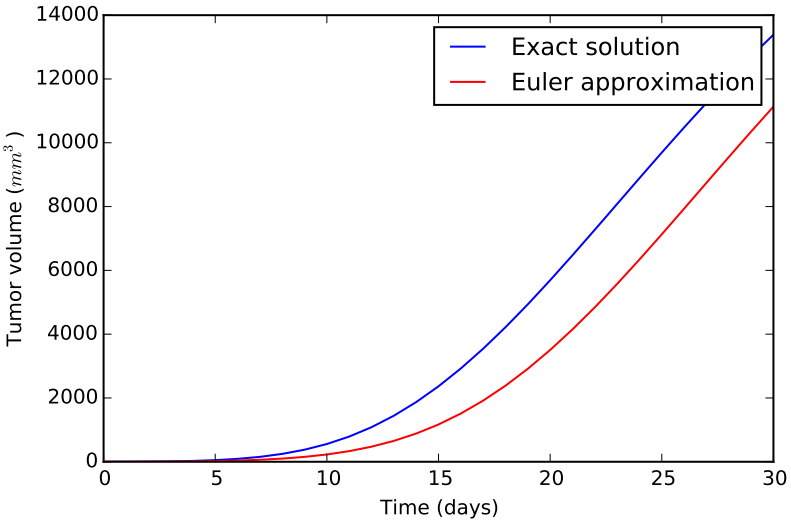
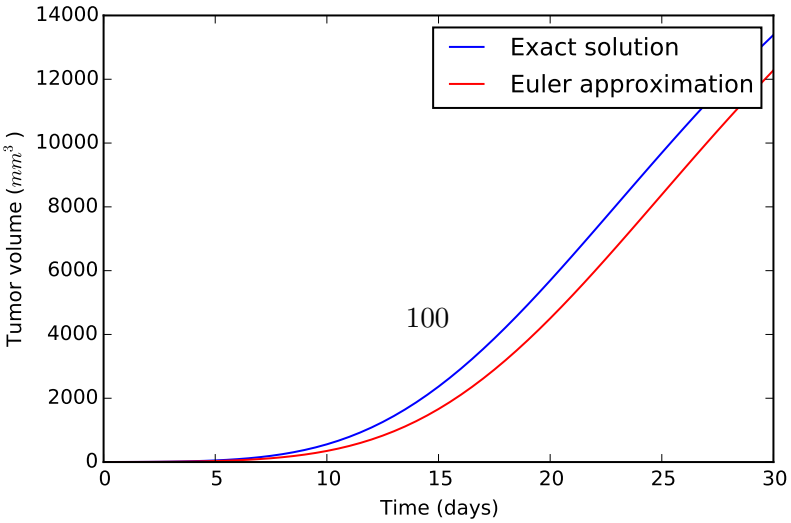


Figure 67: *Euler approximation for* $h = 1$.



Value of the step	Relative error	Calculation time (s)
2	46.2 %	< 0.0001
1	26.4 %	< 0.0001
0.5	14.8 %	< 0.0001

Figure 69: *Relative errors and calculation times for the RK4 approximation depending on the value of the step h*

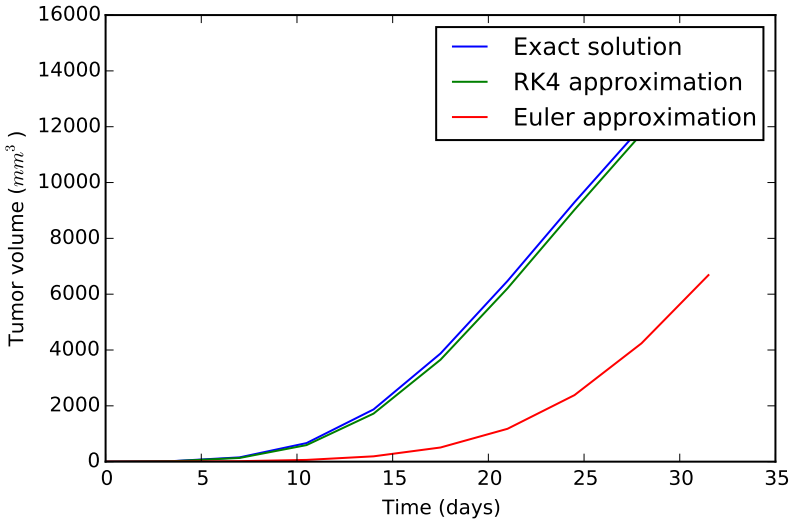


Figure 70: *Euler approximation and RK4 approximation for* $h = 3.5$

Relative error	Bigger step (RK4 / Euler)	Calculation time (s) (RK4 / Euler)
10 %	6.85 / 0.32	< 0.0001 / < 0.0001
1 %	2.2 / 0.03	< 0.0001 / 0.003
0.1 %	1.07 / 0.003	< 0.001 / 0.004

Figure 71: *Bigger steps allowing to get a specific relative error for the two numerical methods and corresponding calculation times*

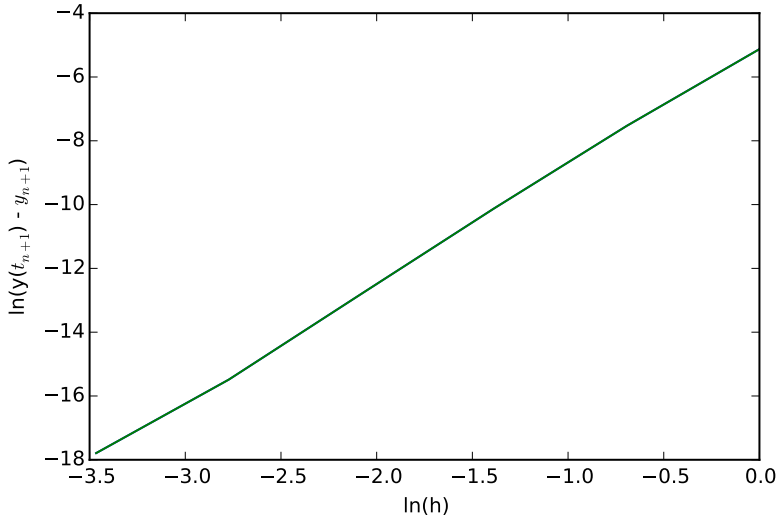


Figure 72: *Evaluation of the order of the RK4 method*

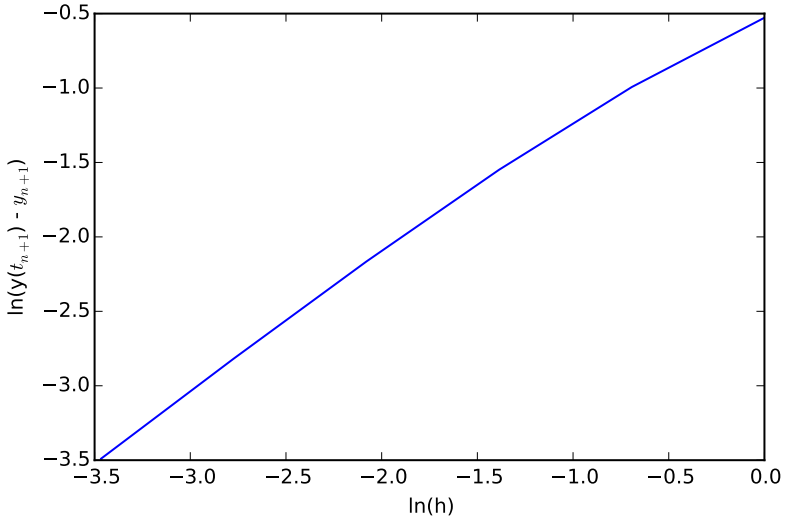


Figure 73: *Evaluation of the order of the Euler method*